MOSIMR : A TIMING SIMULATOR USING SIGNAL PROPAGATION CHARACTERISTICS TO EXPLOIT LATENCY

A Thesis Submitted
In Partial Fulfilment of the Requirements
for the Degree of

MASTER OF TECHNOLOGY

by

SAIBAL BANERJEE

to the

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR
SEPTEMBER, 1985

TO
MY PARENTS

CERTIFICATE

This is to certify that the thesis entitled 'MOSIMR: A TIMING SIMULATOR USING SIGNAL PROPAGATION CHARACTERISTICS TO EXPLOIT LATENCY' submitted by Mr. Saibal Banerjee, for the partial fulfilment of the requirements for the award of M.Tech. degree, has been carried out under my supervision. According to my knowledge, it has not been submitted else where for an award of a degree.

R. Raghuram 4/9/85

(R. Raghuram)
Assistant Professor
Department of Electrical Engineering
Indian Institute of Technology
KANPUR

ACKNOWLEDGEMENTS

Amidst mixed fortunes I have at last come to complete this little piece of work which will ever remain a source of satisfaction as well as inspiration to me.

I express my deep sense of gratitude and appreciation to Dr. K. Raghuram for the splendid guidance and constant encouragement he has offered through the entire endeavour.

I shall ever cherish the invaluable company of my dearest friends Goutam Deb and Biswajit Nandy who never hesitated in lending helping hands whenever needed, at various stages of my thesis work.

Thanks to Mrityunjoy, K.K. Islam, A. Sanyal and host of other friends who have made my stay at I.I.T. Kanpur most gratifying.

Finally, I thank Mr. J.S. Rawat for his excellent service in typing the manuscript.

Sept. 4,1985

Saibal Banerjee

(SAIBAL BANERJEE)

ABSTRACT

Timing and power dissipation are the two important performance indices of an MOS digital circuit. With this in view a program to simulate NMOS/LSI circuit, named MOSIMR, has been developed. It provides the user with initial logic states and timing waveforms of requested node voltages and element powers (as well as total power dissipation) over the simulation period.

The main focus is on minimizing CPU time by exploiting latency of large circuits. The given circuit is decomposed into independent subsystems by node partitioning. A linked data structure of the circuit topology is constructed to order the processing of node partitions in accordance with signal flow. This way a selective trace can be conducted through the circuit to take full advantage of circuit inactivity.

Any change in input travels as a wave in a large circuit. The trace algorithm is capable of dynamically detecting this signal wave, so that only those nodes which are encompassed by the wave profile are analysed on a signal flow path. This results in further exploitation of latency.

With the above scheme of network - ordering, nonlinear Gauss Seidel (GS) relaxation method is employed to evaluate the active nodes only, in each time step. One iteration of GS relaxation, containing one iteration of Newton Raphson, is performed per simulation step.

Results checked against SPICE 2G show reasonable accuracy and remarkable reduction in analysis time with increasing circuit size.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Circuit analysis as a means to verification of a design got a new dimension after the advent of integrated circuit (IC) and its rapid evolution over the past two decades. The traditional paper-and-pencil analysis is a sound method but it imposes severe limitations on the size and type of circuits that can be analysed economically. Large linear (say those containing > 50 elements) [5] or even small nonlinear circuits are seldom analysed exactly. Instead engineers often rely heavily on intuition to obtain approximate solutions. Invariably the final verification of circuit performance is obtained by breadboarding, where there is also scope, though limited, to optimize circuit performance by a trial and error adjustment of element values.

But all these prove grossly inadequate for the analysis, design verification and optimization of integrated circuits, for it is impossible to duplicate the IC with discrete components. Certainly parasitic-effects and element-matching characteristics between integrated devices are not reproducible in discrete form; besides tolerance analysis etc. can not be performed with full freedom as we can scarcely control the device parameters.

A computer program to simulate the circuit can on the other hand build up circuit models of integrated devices to any degree of accuracy and provide control over all the device parameters. In fact it has proven to be the most effective tool for integrated circuit analysis, by which much more accurate results are obtained at a cost which is often a small fraction of the bread boarding cost [5].

With increasing compaction large scale (LSI) and very large scale (VLSI) integrated circuits are being designed and fabricated. A high premium is put on ensuring the correctness of design of these complex circuits before fabrication, because even a minor modification at later stage will result in repeating the expensive and time consuming operations of mask making.

Concerted research efforts have been organised for the last two decades by different groups of people resulting in several general and special purpose circuit simulation programs some of which are enumerated below.

1.1 GENERAL PURPOSE CIRCUIT SIMULATORS:

Simulation program with integrated circuit emphasis (SPICE) is a well known program of this class. Originally developed by Dr. Lawrence Nagel, SPICE was later extensively

modified in version SPICE 1L by E. Cohen and D.O. Pederson and finally got augmented to version SPICE-2G in 1981 by A.R. Newton, D.O. Pederson and A. Sangiovanni-Vincentelli in the Electrical Engineering and Computer Science department of University of California, Berkeley [8].

SPICE is a full fledged circuit simulator using the conventional method of analysis and contains circuit models for semiconductor devices like diode, BJT, FET and MOSFET for application in a wide variety of circuits.

1.1.1 Anatomy of a Conventional Circuit Simulator:

Most general-purpose computer simulation programs have five main stages [5] which are elaborated below:

1.1.1.1 Input Stage:

Here network topology and element specifications are fed in as input data to the program. Preferably this input should be in a format free user language which can be translated by a language compiler inside the simulator, into simulation tables.

Error detection features (e.g. detection of loop of voltage sources, cutsets of current sources) are incorporated with helpful diagnostic messages to make the program user oriented.

Multiple level nesting of circuits, through subscircuit definition by the user, is allowed to facilitate concise description of repeatitive structures.

For a good user friendly program coding for this first stage turns out to be nearly 40% of the whole program.

1.1.1.2 Device model retrieval and replacement stage:

The simulator has a built-in library of device models (e.g. diode, BJT, FET, MOSFET). New models defined by the user, as subcircuits, are also stored efficiently.

The user has control over all the device parameters in the model. Any parameter unspecified by the user is assigned a default value by the program.

An efficient retrieval scheme is kept for quick replacement of a device, when it appears within a circuit, by its equivalent model.

The first two stages simulate the circuit in question and get it ready for analysis.

A general purpose simulator can do three kinds of analysis i) DC-bias point analysis on linear and and non-linear circuits, ii) AC small signal analysis on linear circuits, iii) Transient analysis on linear and nonlinear circuits.

1.1.1.3 Equation formulation stage:

Here circuit equations are framed according to the kind of analysis to be done, as summarised below -

Kind of Analysis	Equations formulated
i) Linear AC (frequency domain analysis)	Linear nodal equations linear hybrid equations.
ii) Linear transient (time domain analysis)	Linear state equations
iii) Nonlinear d.c. (resistive analysis)	Nonlinear nodal equations, nonlinear hybrid equations.
iv) Nonlinear transient (dynamic analysis)	Discretised nonlinear nodal equations Discretised nonlinear hybrid equations Nonlinear state equations

The equations generally in the algebraic-differential equations form represent the dynamical system either in the frequency or time domain. Prior to framing the equation for any kind of analysis however a DC operating point calculation is performed to set the initial condition.

1.1.1.4 Numerical solution stage:

The equations framed in previous stage are solved here employing different numerical methods depending on the types of equations, as tabulated below -

Type of equations	Method of solution employed
i) Linear nodal/hybrid equations	Gaussian elimination, LU - decomposition
ii) Linear state equations	By numerical evaluation of e^{AT} (state transition matrix), Numerical integration (e.g. Backward Euler (BE), Tropezoidal rule (TR), Gear's variable order method (GE))
iii) Nonlinear nodal/hybrid equations	Newton Raphson (NR) method, Piecewise linear method (for getting driving point and transfer characteristic curves)
iv) Nonlinear state equations	Explicit/Implicit numerical integration

1.1.1.5 Output stage:

This final stage gives the results of analysis in any of the forms, summarised below, as requested by the user, depending on the kind of analysis.

Kind of analysis	Forms of result obtained at output stage
i) Linear AC(frequency domain analysis)	Frequency response,distortion analysis
ii) Linear transient (time domain analysis)	Step response, pulse response, periodic input response

iii) Nonlinear DC (resistive analysis)	Operating point calculation (DC bias), Driving point and transfer characteristic plots.
iv) Nonlinear transient (dynamical analysis)	Transient response, periodic response for different waveforms.

1.1.2 Disadvantages faced in Conventional Simulators:

The conventional circuit simulators are versatile and accurate but they are cost effective for circuits containing only few hundred elements or less. For example SPICE 1L can analyse circuits having at most 100 elements. In the quest for versatility SPICE incurs tremendous overhead (sec. A.1) in the first two stages (sec. 1.1.1).

Sparse matrix method of solving the simultaneous eqns. in SPICE exploits sparseness of large circuits to save analysis time to some extent. But it is unable to detect and exploit latency (A.3.5) of large circuits. Besides SPICE operates in circuit level only, whereas in many cases it is possible to make with less accuracy than that provided by a full blown circuit simulator. On the whole a conventional circuit simulator has proved to be expensive in terms of CPU time for the analysis of large circuits and has led to the development of special purpose simulators.

1.2 SPECIAL PURPOSE SIMULATORS:

The special purpose simulators are meant for large circuits (e.g. LSI and VLSI) with dedicated use to avoid unnecessary overheads (Sec. A.1). The main aim in the design of such simulators is to improve upon CPU time.

Simulators such as MOTIS [10], SPLICE, DIANA, MACRO [11], all special purpose simulators, in their quest for speed have rejected one or more of the principal features of conventional simulators.

Fig. 1.1 shows the relative positions of these simulators on a speed/accuracy scale [15]. On one extreme are the

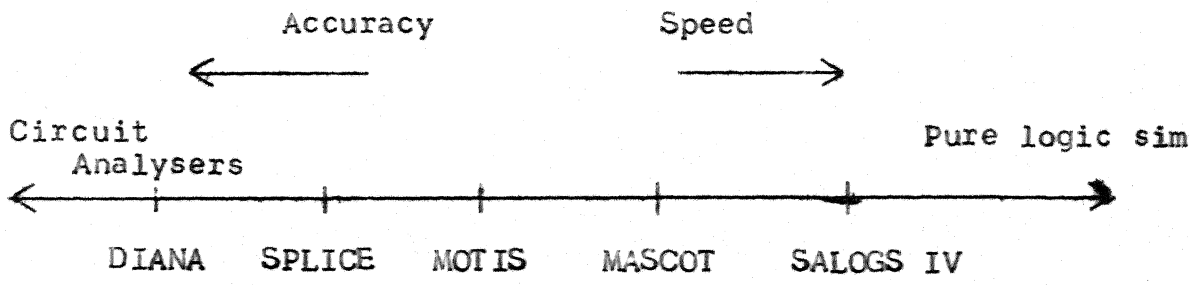


Fig. 1.1: Spectrum of simulators

conventional circuit simulators and on the other end are the logic simulators. Bridging the gap between these are timing simulators like MOTIS and hybrid simulators (i.e. both logic and timing waveforms simulated) like SPLICE.

1.2.1 Logic Simulation:

A foremost example of logic simulator is Logic Analyser for Maintenance Planning (LAMP) [9], a digital logic-simulation program developed at Bell Telephone Laboratories around 1970 with elaborate test generation capability for fault diagnosis.

LAMP simulates the circuit at gate level (NOT, NAND, NOR etc.) functional level (register, memory cell etc.) or at hybrid level. This sort of macromodeling helps to speed up the simulator (Sec. A.2).

LAMP also applies controllability and observability relation among functional nodes (compare with Sec. A.3.9) to yield a directed graph representation of the digital circuit. Any loop within the graph due to feed backs in the circuit are 'broken' by addition of control and observation at appropriate places to remove conflict (compare with Sec. A.3.10) and build up a link graph (Sec. A.3.9.5).

A link graph gives an ordering of the nodes (Sec. A.3.9.3) where a 'higher order' node can be computed independently of any 'lower order' node(s).

This way LAMP implements a very important concept of partitioning and ordering a network which helps to bring down CPU time remarkably (Sec. 1.2.2.2).

Also, the final link graph structure establishes a precedence order of signal flow and it becomes possible to conduct a selective trace along the signal flow path and exploit lateney (Sec. A.3.5) of large circuits. With all these new concepts brought in LAMP is capable to simulate circuits containing 52000 gates, economically on a IBM 370 model 168 machine [9].

SALOGS IV is a logic-timing simulator which models logic-gate inertial effects as pure delays rather than producing merely the logic levels. But it does not provide the analog waveforms. It is some ten thousand times faster than a circuit simulator and is capable of analysing systems of many thousands of gates [15]. MASCOT also models inertia as pure delays but gives an analog slope for rising and falling edges [15].

1.2.2 Timing Simulation:

Besides logic simulation timing analysis has been found to be of great interest in digital circuits. It is here that the user can get a more accurate picture of the profiles of different circuit variables (e.g. node voltages, device currents and power dissipation etc.) against time, rather than the simplistic picture of logic levels plus discrete propagation delays provided by the logic simulators.

Timing analysis for large circuits is very expensive with conventional circuit simulators. Research efforts in this direction have yielded special purpose simulators which incorporate many time saving features of logic simulators like i) Minimising overheads ii) Macromodeling, iii) Network partitioning and ordering iv) Exploitation of circuit latency etc. This way these special purpose simulators have been able to achieve on an average one order of magnitude increase in speed over conventional simulators, while maintaining accuracy of results comparable to that of standard circuit simulators.

Thus timing simulation stands mid way between logic simulation and standard circuit analysis and proves to be the best compromise between speed and accuracy of analysis.

1.2.2.1 Timing simulation techniques:

There are three kinds of timing simulators depending on the techniques they employ. All the three kinds use relaxation method,but differently and at different stages of the algorithm for transient analysis (Fig. 1.2). The general relaxation method of analysing a system of algebraic-differential equations have been elaborated in Sec. A.3 of Appendix A. Representative examples of these three classes of simulators for time domain analysis are 1) MOTIS - a timing simulator

2) SPLICE - a mixed mode simulator performing iterated timing analysis (ITA) and 3) RELAX - using waveform relaxation method to perform timing analysis.

Timing simulation done in MOTIS as well as iterated timing simulation done in SPLICE apply Gauss-Seidel (GS) method. (Sec. A.3.2) in the nonlinear algebraic equation stage of circuit analysis, thus carrying out a Gauss-Seidel-Newton (i.e. Newton Raphson (NR) iterations included within each G-S iteration) method to decouple and solve the circuit equations obtained by discretisation. In MOTIS only single iteration of G-S and NR are performed in each time step (Sec. A.3.3) whereas SPLICE 1.6 iterates these to convergence [12].

Timing errors (refer Sec. 4.1.3 and Sec. A.3.4) which are incurred in timing simulators for non-unidirectional circuits can be avoided in ITA. But large number of strong feedbacks often slow down the ITA process considerably as too many iterations have to be performed before convergence could be reached.

Unlike MOTIS and SPLICE, RELAX applies the relaxation technique right in the differential equation stage of analysis. Naturally the unknowns here are waveforms (elements of a function space) rather than real variables. The trial

solution to start with is the guessed waveform of the unknown over the entire period of integration O to T. Accurate comparison between waveform-relaxation (WR) method, as it is called and ITA have not been carried out yet [11], [12]. But both have proven to be very accurate and having guaranteed convergence properties when applied to the analysis of MOS circuits.

1.2.2.2 Comparison with general circuit simulator:

All the three classes of special purpose timing simulators out perform the general purpose circuit simulators speed wise, mainly due to the following reasons:

1. Circuit decomposition by assignment partitioning (effected by the application of relaxation method) greatly reduces the complexity of analysis is (Sec. A.3.1). A circuit which is not partitioned has complexity of analysis proportional to n^m where n is the size (i.e. no. of unknown variables) of the circuit and m is a number depending on the sparsity of the circuit. m is small (large) if the number of interactions among unknown variables is small (large). m is usually between 1.2 and 2 [11].

   But with decomposition, the complexity of analysis is usually proportional to n. Thus circuit decomposition is a major factor in reducing analysis time, specially in large circuits.

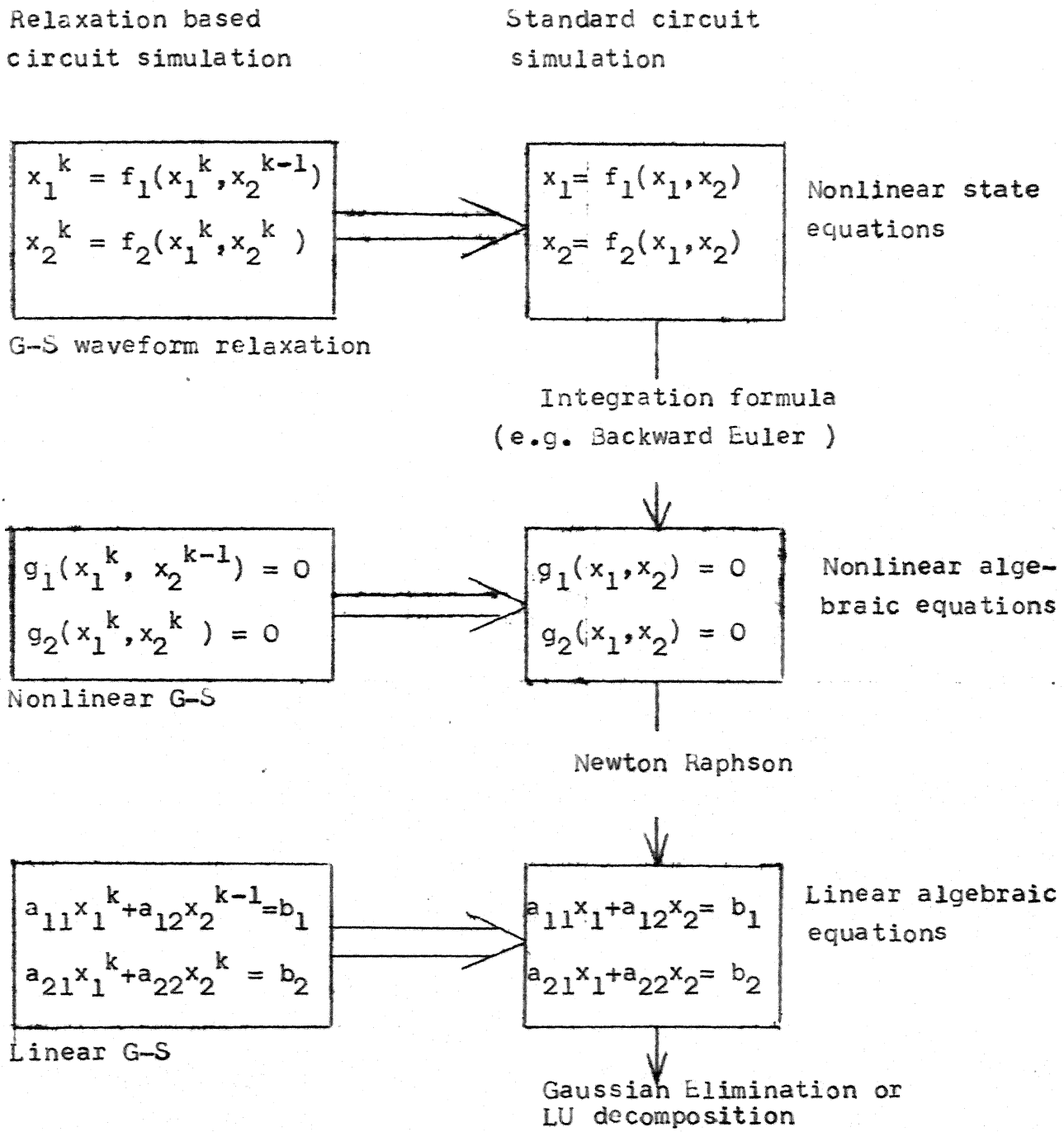


Fig. 1.2: Parallel between standard circuit simulation technique and relaxation based techniques.

2. Circuit partitioning breaks up the whole circuit into subsystem blocks which can be analysed independently. A scheduler built on the same line as in logic simulators, order the computation that helps to speed up convergence if G-S relaxation is in use (Sec. A.3.4). As well as it builds up a link graph structure of the subsystems and allows most naturally to schedule a method to exploit latency and avoid unnecessary computation of subsystems which are inactive in a particular time step. As a matter of fact relaxation methods are ideally suited to exploit the latency (Sec. A.3.5) of the circuit under analysis.

3. Finally the special purpose programs built for specialised applications can avoid overheads (Sec. A.1) incurred by general purpose simulator. The speed plus accuracy achieved by timing simulators is possible because they punch the scheduling techniques of logic simulators for ordering the computations of subsystem blocks, with the full fledged circuit analysis of conventional simulators applied to the computation of individual subsystem.

1.3 TIME DOMAIN ANALYSIS OF MOS LSI CIRCUITS:

The computation cost for time-domain simulation of large circuits is prohibitive. The cost lies principally in

function evaluations for the Jacobian. Again the whole of the Jacobian (coefficient matrix) must be memory resident if equation solution efficiency is not to be impaired [15]. Despite falling cost of computing memory, this latter factor remains a problem because of the demands in LSI and VLSI to analyse larger and larger circuits. Indeed, the practical limit for analysis at present is probably a circuit of about thousand nodes, but this is much smaller than the current LSI circuits.

Fig. 1.3 shows the speed of analysis vs. circuit size curve for transient analysis of nonlinear circuits with a

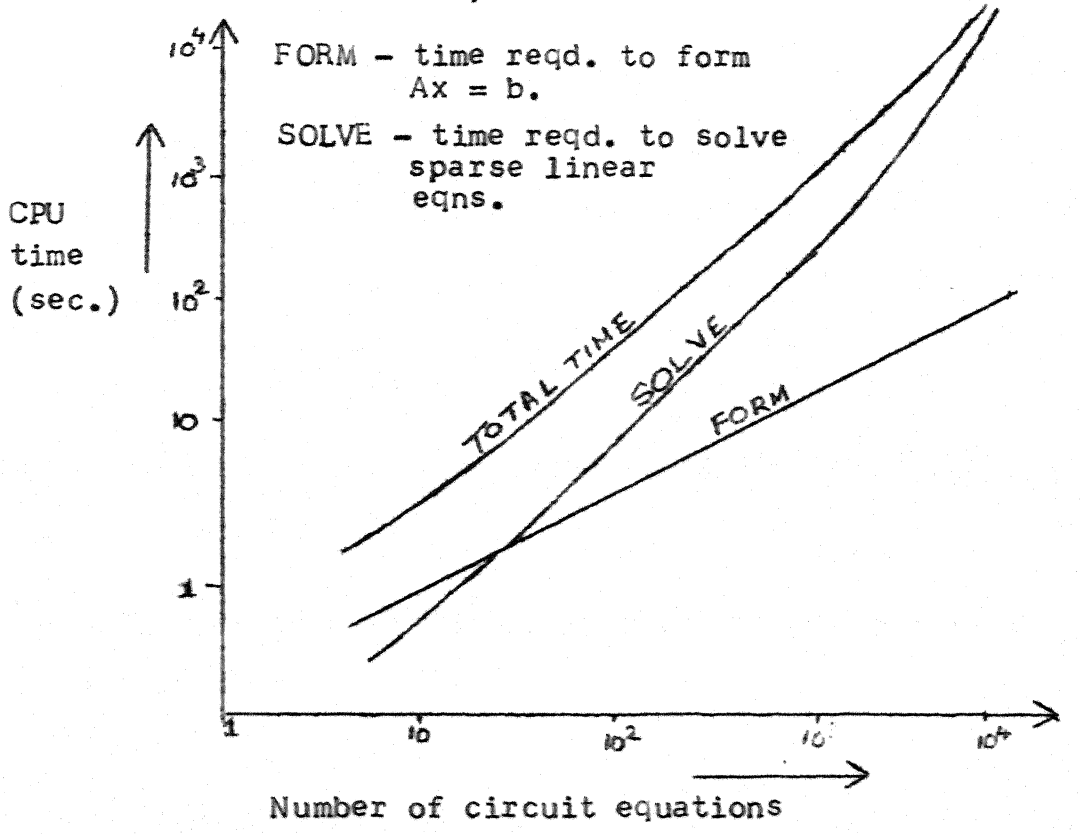


Fig. 1.3: Transient analysis time for circuits of increasing size (i.e. no. of unknown variables)

conventional circuit simulator. FORM, which includes time required for discretisation and linearisation, is found to dominate over SOLVE, which is the time for solution of the set of linear equations, only in case of small circuits (having < 50 nodes). Special purpose simulators have come up to face the problem of transient analysis of large nonlinear circuits which appears so in surmountable with standard circuit simulators.

MOSIMR is such a special purpose simulator designed with following aims and implemented in FORTRAN 10 on a DEC 1090 system.

1.4 AIMS IN DEVELOPING MOSIMR:

Aim here is to develop a special purpose transient analysis program for digital NMOS/LSI circuits. Speed and power dissipation are the most important factors considered for the evaluation of an MOS LSI design. MOSIMR is meant to provide timing waveforms of voltages at nodes of interest and power dissipation in elements of interest, with reasonable accuracy.

Focus is on improving speed of analysis and minimising storage requirement, to make analysis of large circuits with MOSIMR economical.

The program is to compute initial condition based on a full-fledged logic simulation of the given digital circuit.

Hence it can be made useful as a mixed mode simulator that can perform as a logic simulator at t=0 and as a timing simulator subsequently.

In the quest for speed, new techniques of dynamic detection of the traveling signal-wave profile are to be envisaged in program implementation, for fuller exploitation of latency. This also makes the program a useful tool as a signal tracer in large circuits.

The program is also meant to be interactive. The course of simulation will be controlled and guided by the user to a large extent.

1.5 THESIS OUTLINE:

The following chapters dealing with the design, implementation and performance evaluation of the NMOS timing simulator, MOSIMR, are organised as follows:

Chapter 2 deals with defining the circuit analysis problem and formulating the theoretical scheme for solution, as followed in MOSIMR. It also mentions the assumptions made to simplify the solution scheme.

Chapter 3 gives the details of program implementation in terms of flow charts and explains the function of each program module. A glossary of variables and arrays used in the program has been appended here to make the flow charts understandable.

Chapter 4 gives the results obtained by analysing a few typical circuit configurations and compares the analysis time and accuracy of the results obtained by MOSIMR with that of the results obtained by the standard general purpose circuit simulator SPICE 2G.

Chapter 5 draws conclusion by pointing out credits and limitations of the program MOSIMR, its various possible applications and finally suggesting scopes of modification for its betterment.

Appendix A is there to clarify some important terms and concepts through definition, brief explanation/derivation and illustrative examples.

Appendix B gives a complete manual for MOSIMR, to help and guide the user in building input data files for the simulator. It also makes the user aware and cautious of all the idiosyncrasies of the program.

CHAPTER 2

THEORETICAL FRAMEWORK OF MOSIMR

2.1 DEFINITION OF THE PROBLEM:

The digital NMOS circuit to be analysed is described in terms of transistors/gates/functional blocks like pass transistor/logic gate/latch. Each of these constitutes an element (Sec. A.2). Interconnection of these elements are the functional nodes (or simply nodes).

An n-node circuit having initial state v_o is a dynamical system characterised by a set of mixed-implicit algebraic - differential equations of the form

$$F(\dot{y}, y, u) = 0 \qquad (2.1a)$$

$$v(o) = v_o \qquad (2.1b)$$

where,

$y(t) \in R^n$ is the vector of unknown variables, namely the n node voltages.

$\dot{y}(t) \in R^n$ is the time derivative of y at time t

$u(t) \in R^r$ is the vector of r input variables at time t

$y_o \in R^n$ is the vector of initial values of y(t) at time t.

$F: R^n \times R^n \times R^r \rightarrow R^n$ is a continuous function.

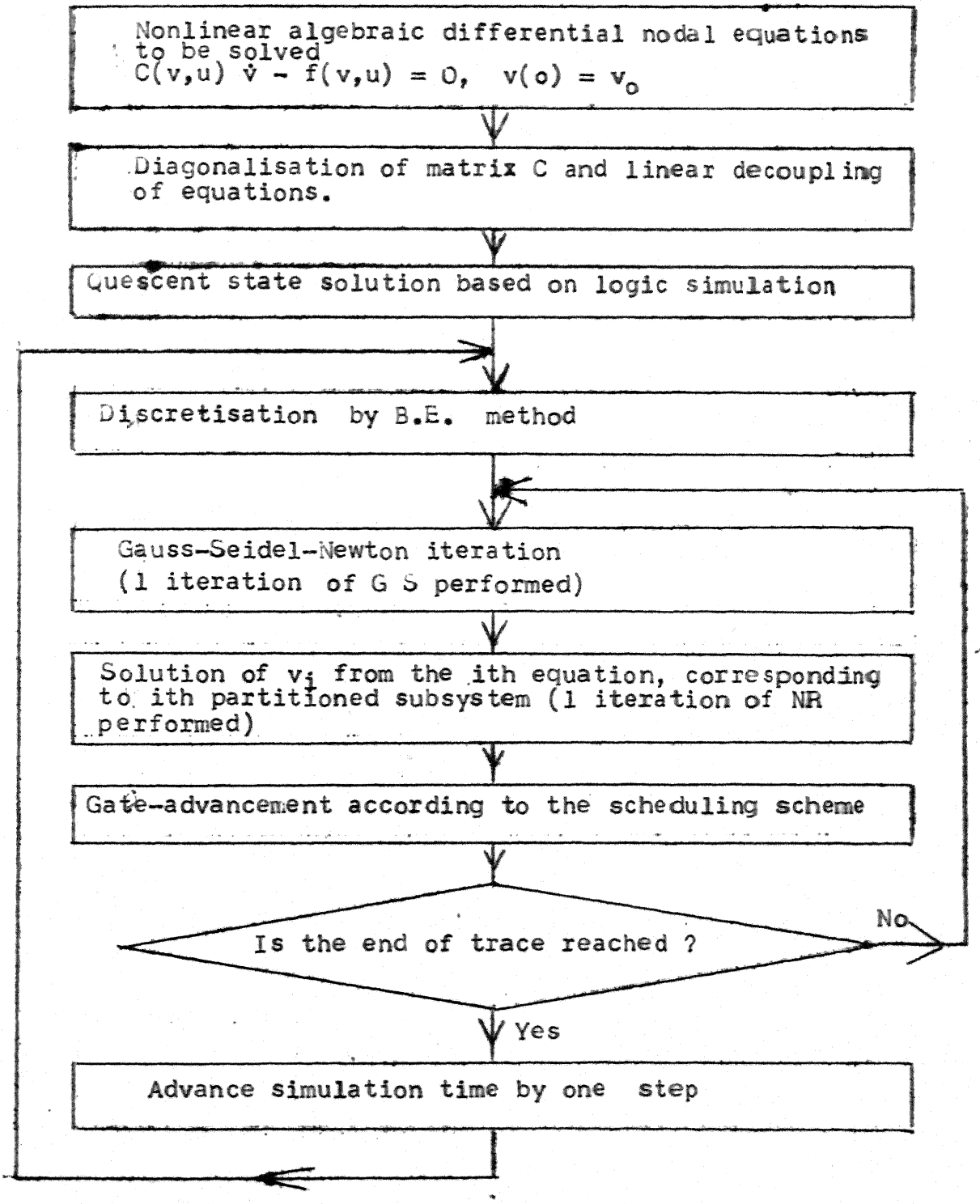


Fig. 2.3: General scheme of solution followed in MOSIMR

Thus the problem of time domain analysis of NMOS LSI circuits taken up by MOSIMR boils down to the solution of the dynamic equation (2.1a) with the help of initial conditions of eqn. (2.1b).

The nodal equations for a MOS circuit with n-nodes can be written as follows:

$$C(v,u)\dot{v} = f(v,u) \qquad (2.2a)$$

with initial condition

$$v(o) = v_o \qquad (2.2b)$$

where $v \in R^n$ is the vector of all unknown node voltages and v_o, the given initial values of v.

$u \in R^r$ is the vector of all inputs

$f: R^n \times R^r \rightarrow R^n$ is a continuous function each component of which represents the net sum of currents charging the capacitor at each node due to the conductors and the controlled current sources (MOS devices) connected to the node.

$C: R^n \times R^r \longrightarrow R^{nxn}$ is a symmetric matrix in which $C_{ii}(v,u)$ is the sum of capacitances of all the capacitors connected to node i.

$-C_{ij}(v,u)$; $i \neq j$ is the total capacitance between nodes i and j.

We can rewrite equations (2.2) as follows,

$$C(v,u)\dot{v} - f(v,u) = 0 \qquad (2.3a)$$

$$v(o) = v_o \qquad (2.3b)$$

which are of same form as eqns. (2.1) and describe the dynamic system of the NMOS LSI circuit which MOSIMR seeks to analyze.

2.2 ASSUMPTIONS:

i) The circuit is considered to be at steady state at t = 0. Hence the initial state v_o is obtained by a dc solution of the circuit, prior to its time domain analysis.

ii) During computation of $v_i(t)$, except the voltage at node i, all other nodes are considered to be constant.

This assumption has vital implication in terms of floating capacitors and node partitioning.

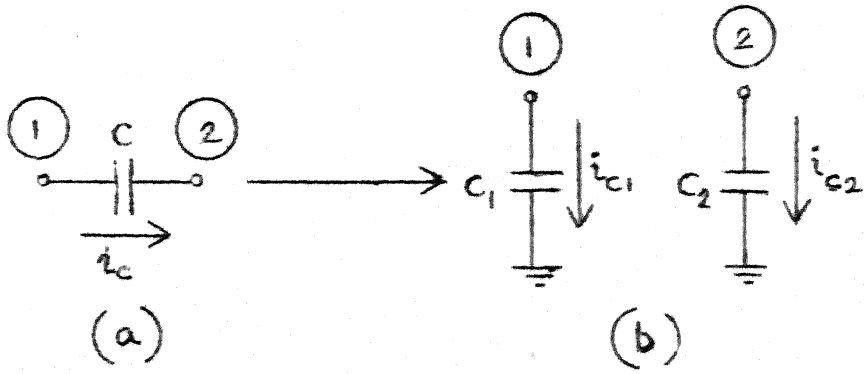


Fig. 2.1: Replacement of a floating capacitor by equivalent grounded capacitances.

Let us consider the floating capacitor between nodes (1) and (2) in Fig. 2.1a. i_c is the current, sourced by node 1 and sinked by node 2, which flows through the capacitor c.

Hence $i_c = C \frac{d}{dt} (V(1)-V(2))$ (2.4)

When we solve for V(1) we assume V(2) to be constant (equal to VC say).

So the current charging the node capacitor at node 1 is

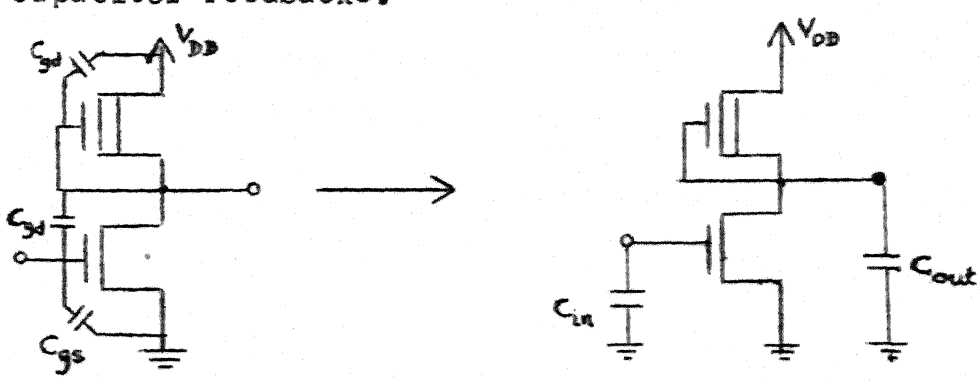
$$ic_1 = C \frac{d}{dt} (V(1)-VC) = C \frac{dV(1)}{dt}$$

Similarly the node capacitor current at node 2 is

$$ic_2 = -C \frac{dV(2)}{dt}$$

which is equivalent to having grounded capacitors of value C at nodes 1 and 2 with currents ic_1 and ic_2 charging them as shown in Fig. 2.1b.

This simple reduction of floating capacitors into equivalent grounded capacitors helps to reflect floating device capacitances (e.g. C_{gd}) at the input/output nodes of gates as shown in Fig. 2.2 and free the circuit from floating capacitor feedbacks.



$C_{in} = C_{gs} + C_{gd}$; $C_{out} = 2C_{gd}$

Fig. 2.2: Reflecting floating device capacitances at gate input/output nodes.

In the way as shown in Fig. 2.2 all floating capacitances are replaced by equivalent grounded capacitances to diagonalise and decouple eqn. (2.3a) as follows:

$$\begin{aligned}
\dot{v}_1 &= \frac{1}{C_{11}} f_1 (v_1, d_1, u) \\
\dot{v}_2 &= \frac{1}{C_{22}} f_2 (v_2, d_2, u) \\
&\vdots \\
\dot{v}_i &= \frac{1}{C_{ii}} f_i (v_i, d_i, u) \\
&\vdots \\
\dot{v}_n &= \frac{1}{C_{nn}} f_n (v_n, d_n, u)
\end{aligned} \qquad (2.5)$$

Except v_i (the node voltage to be solved from the ith eqn.) in eqns. (2.5) all other node voltages involved in the ith equation are considered constant (by assumption (ii)) and have been included in the vector d_i.

2.2.1 Validity of the Assumptions:

Assumption (i) remains valid for circuits which are in steady state at t=0 which means currents through 'on' pass transistors and node capacitor currents must all be zero initially. But any attempt to analyse circuit which is in a transient state will make assumption (i) invalid.

Assumption (ii) becomes clearly invalid if time step of simulation is too large. For its validity an upper bound is imposed on the simulation step size.

2.3 GENERAL SCHEME OF SOLUTION:

The flow diagram in Fig. 2.3 sketches the general scheme followed by MOSIMR in solving eqn. (2.3). As apparent from the flow diagram the scheme is similar to that followed by a noniterative timing simulator (Sec. 1.2.2.1). Relaxation method employed is one of Gauss Seidal iteration at the discretised nonlinear equation level.

But new ideas have been introduced in scheduling selective trace (Sec. A.3.7) based on signal flow to make a fuller exploitation of latency.

The general scheme after it has reached the 3rd stage shown in Fig. 2.3, consists of three major processes namely i) Discretisation ii) Assignment partitioning iii) relaxation process.

2.3.1 Discretisation:

Discretising the ith eqn. in eqns. (2.5) by Backward Euler method,

$$v_i^{t+1} = v_i^t + \frac{h}{C_{ii}} f_i(v_i, d_i, u) \Big|_{t+1} \qquad (2.6)$$

where i = 1 to n and h = time step of integration.

2.3.2 Gauss-Seidel-Newton Method:

This process involves application of Gauss Seidel iteration at the non-linear equation level with an NR interation nested within each GS iteration.

2.3.2.1 Assignment partitioning:

This is related to a system of simultaneous equations. Each unknown variable is assigned to an equation in which it is involved (Sec. A.3.1). However no two variables are assigned to the same equation thus giving n disjoint independently solvable equations.

Physically this amounts to partitioning the circuit into n independent subsystems each having an unknown node to be evaluated.

From graph theoretic point of view assignment partitioning will be recognised as breaking of the MSC subgraphs to resolve conflicts (Sec. 2.5).

Assignment partitioning of eqn. (2.6) is performed using nonlinear Gauss-Seidel algorithm as follows;

$$v_i^{t+1} = v_i^{t} + \frac{h}{C_{ii}} f_i(v_i^{t+1}, d_{u_i}^{t}, d_{l_i}^{t+1}, u^{t+1}) \qquad (2.7)$$

where i = 1 to n

v_i is the unknown voltage assigned to the ith partitioned subsystem

$$d_{l_i} \subseteq S_{l_i}$$
$$d_{u_i} \subseteq S_{2_i}$$

where S_{2_i} is the set of nodes to be evaluated in time step t+1, after node i.

and S_{l_i} is the set of nodes already evaluated in t+1, before node i.

S_1 and S_2 depend on the scheduling algorithm that determines the actual sequence in which the equations are solved.

Except v_i all other variables have been included in d_{l_i} and d_{u_i} in eqn. (2.7). These are the decoupling inputs of the ith subsystem (Sec. A.3.1) and by virtue of assumption (ii) in Sec. 2.2 these are considered constant, while v_i is computed from 2.7 in the time step t+1.

2.3.2.2 Newton-Raphson (NR) method:

In order to solve v_i^{t+1} from eqn. (2.7) NR method is applied to linearise the nonlinear current function f_i as follows:

From eqn. (2.7) we have

$$F_i(v_i^{t+1}) = v_i^{t+1} - v_i^t - \frac{h}{C_{ii}} f_i(v_i^{t+1}, d_{u_i}^t, d_{l_i}^{t+1}, u^{t+1}) \quad (2.8)$$

Hence root of the eqn.

$$F_i\,(v_i^{t+1}) = 0 \tag{2.9}$$

is the required solution of v_i in time step t+1. Now

$$F_i'(v_i^{t+1}) = 1 - \frac{h}{C_{ii}} f_i'(v_i^{t+1}, d_{u_i}^{t}, d_{l_i}^{t+1}, u^{t+1}) \tag{2.10}$$

where $F'(v_i^{t+1})$ is the derivative of F, w.r.to v_i^{t+1}
and f' is the derivative of f, w.r. to v_i^{t+1}.

Hence by N.R. formula

$$F_i(v_i^{t+1})^1 = F_i(v_i^{t+1})^0 + \left\{(v_i^{t+1})^1 - (v_i^{t+1})^0\right\} F_i'(v_i^{t+1})\Big|_{(v_i^{t+1})^0} \tag{2.11}$$

where $(v_i^{t+1})^1$ is the solution
and $(v_i^{t+1})^0$ is the trial value

Taking the previous time step value of v_i as the trial value and the value of v_i in this time step, as the solution;

$$(v_i^{t+1})^0 = v_i^{t} \text{ and } (v_i^{t+1})^1 = v_i^{t+1} \tag{2.12}$$

Hence from eqn. (2.11), and (2.12) we have

$$F_i(v_i^{t+1}) = F_i(v_i^{t}) + (v_i^{t+1} - v_i^{t})\, F_i'(v_i^{t+1})\Big|_{v_i^{t}} \tag{2.13}$$

Let $\Delta v_i = v_i^{t+1} - v_i^{t}$

Then using eqns. (2.8), (2.9) and (2.10) eqn. (2.13) gives

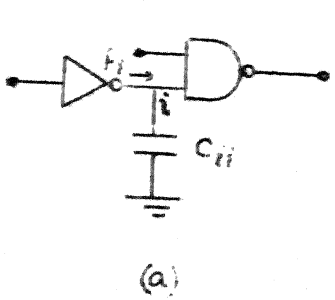
$$\Delta v_i = \frac{f_i(v_i^{t},\ d_{l_i}^{t+1},\ d_{u_i}^{t},\ u^{t+1})}{(C_{ii}/h) - f_i'(v_i^{t},\ d_{l_i}^{t+1},\ d_{u_i}^{t},\ u^{t+1})} \qquad (2.14)$$

where f_i is a nonlinear function of several node voltages in general and denotes the net current flowing into the node capacitance C_{ii}. f_i' is the dynamic slope of the node capacitance current vs. node voltage and hence denotes output conductance at node i. It is equal to the algebraic sum of the conductances of all devices having output at node i. Equation (2.14) is used by MOSIMR to compute v_i^{t+1} from v_i^{t} (refer Sec. 2.4).

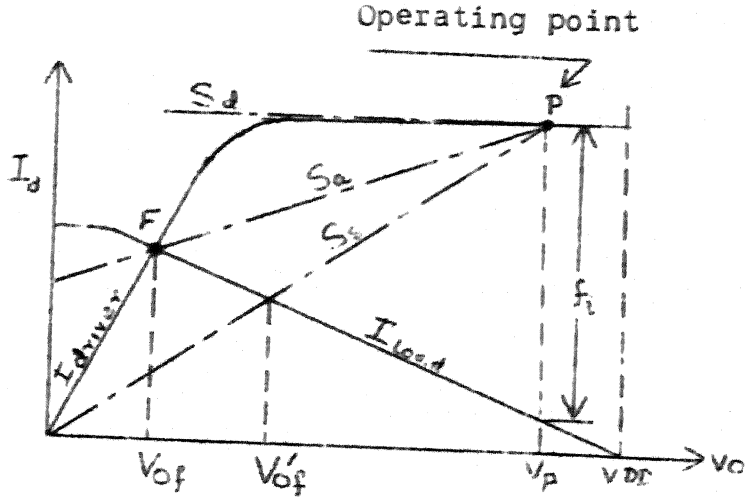
2.4 VALIDITY OF THE SOLUTION:

Fig. 2.4 shows a simple case where i is the node to be computed with C_{ii} as the corresponding node capacitance and f_i, the capacitor charging current flowing in from the inverter output.

For the solution in eqn. (2.14) obtained by a single iteration of N.R. to be asymptotically correct, f_i' the slope used in eqn. (2.14), must be redefined.



(a)



f_i = I driver - I load

(b)

Figure 2.4

In the limit $h \to 0$ eqn. (2.14) reduces to $\Delta V_i = \frac{h f_i}{C_{ii}}$ which is merely a difference form of eqn. (2.2a).

In the other limit $h \to \infty$ eqn. (2.14) reduces to

$$\Delta V_i = f_i / f'_i \qquad (2.15)$$

Fig. 2.4b shows the V-I characteristics of the load and driver devices of the inverter in Fig. 2.4a. Approximating the driver curve by a line joining the operating point P and the final point F we have f_i linearly varying with V_o with a slope

$$GG = \frac{f_i}{V_P - V_{of}}$$

Redefining f' as GG, in equation (2.15), we get

$$\Delta V_i = V_p - V_{of} \quad \text{as } h \to \infty$$

Thus with the new definition of f' asymptotic validity of eqn. (2.14) can be ensured, though large errors in intermediate points can not be avoided. V_{of} is not easily computable hence in MOTIS instead, the line joining the operating point P with the origin (having a static slope S_s in Fig. 2.4b) is used to approximate the driver curve. This incurs greater error in intermediate points and results in some asymptotic error also as it ends up with the final voltage V'_{of} (refer Fig. 2.4b) instead of V_{of}.

MOSIMR on the other hand uses the dynamic slope (S_d in Fig. 2.4b) of the driver curve. This gives less error in intermediate points than the method using static slope and is specially accurate for small values of h. However, it has no asymptotic correctness and may lead to total instability for large h. Therefore for eqn. (2.14) to remain valid, an upper-bound over h has to be imposed.

2.5 NODE PARTITIONING:

Node partitioning as done by Gauss Seidel method in MOSIMR is based on controllability considerations (Sec. A.3.11). In ordinary cases the partitioned subsystem consists of the concerned node and one control edge (Sec. A.3.9) directing to it. Physically this corresponds to a gate with the concerned node as its output node.

MOSIMR does not allow wired connection of outputs and wherever there are more than one controls fanning into a a node it is due to presence of pass transistors (as at node N in Fig. 2.5c) producing conflict (Sec. A.3.9.4). MOSIMR resolves such conflicts by forming a fan-in table (A.3.11). A partitioned subsystem centred around such a conflicting node contains the concerned node and all the fan-in elements (A.3.11.1) of that node, which have to be evaluated in order to evaluate the node in a particular time step.

2.5.1 Tackling MSC Subgraphs of Different Kinds (Sec. A.3.9.3):

Figs. 2.5a - 2.5c show the typical cases where digital circuits form MSC subgraphs. In each case MOSIMR resolves the conflict at node N as follows:

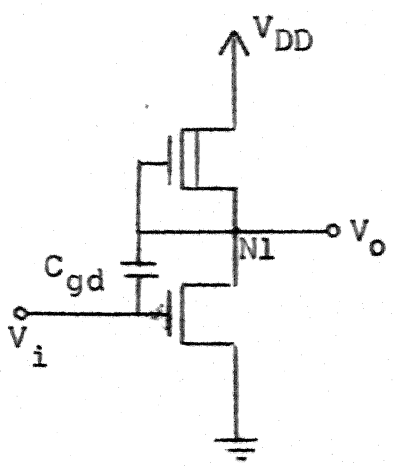


Fig. 2.5(a)

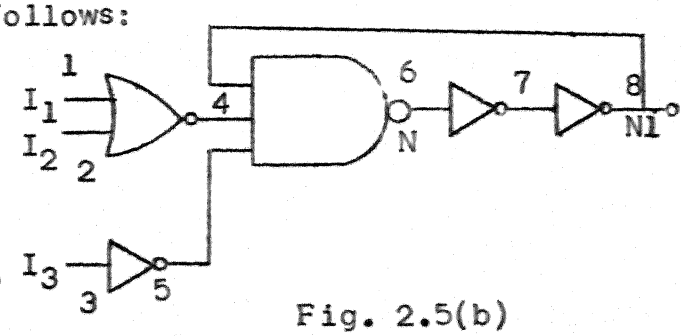


Fig. 2.5(b)

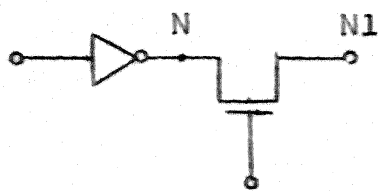


Fig. 2.5(c)

In Fig. 2.5a presence of the floating capacitor C_{gd} causes N to depend on a lower order (Sec. A.3.9.3) node N1. But in MOSIMR these two nodes are decoupled by removing the floating capacitor C_{gd} and replacing by equivalent grounded capacitors reflected at the input and output. This resolves the conflict in computing N.

In Fig. 2.5b computation of N depends on the lower order node N1, due to a feed back. But MOSIMR uses the voltage value of N1 of previous time step t to compute N at t+1 (Sec. A.3.4). This introduces a minimal error of 1 time step, generally, when using G-S method.

One limitation of MOSIMR will be pointed out in this context. As the scheduling algorithm employed in MOSIMR is unable to distinguish between a forward path and a feedback path, this method of using the previous time step value of the controlling node, when it is still to be evaluated in the present time step, is applied blindly. In Fig. 2.5b for example if inputs are scanned in the order I_1, I_2, I_3 and a change is monitored in I_2 the scheduler conducts a trace through nodes 4,6,7 and 8. When come to evaluate 6 i.e. node N at t+1 it is found to depend on nodes 4,8 and 5. Of the three controlling nodes both 8 and 5 are unevaluated at t+1. Of these two 8 can not be evaluated unless 6 is

computed. However, 5 can be computed independent of 6 as it does not belong to a feedback path. It is therefore physically possible to compute node 5 at t+1 before 6 is evaluated. But as the scheduler in MOSIMR is unable to tell between a feedback path (such as node 8 in Fig. 2.5b) and a normal forward path (node 5 in Fig. 2.5b) both nodes are treated equally by taking their previous time step values to compute 6, thus incurring further timing errors.

In Fig. 2.5c node N depends on the lower order node N1 if the bidirectional switch (the pass transistor) is 'on'. The ensuing conflict is resolved by MOSIMR during node partitioning process. The partitioned subsystem assigned to evaluate N includes the inverter as well as the pass transistor (with node N1)as fan-in elements (sec. A.3.11).

2.6 NETWORK ORDERING:

Node computation ordering is based on observability relations of nodes (Sec. A.3.9). This relationship is built in the program MOSIMR in form of fan-out table or linked element list [19] from the circuit description (Sec. 3.2). A selective trace is conducted through the circuit along the signal flow paths with the help of a scheduler. This way inactive/latent parts of the circuit are excluded from analysis.

The node computation sequence is determined by a circular queue [19] which acts as a scheduler (Sec. A.3.8) in MOSIMR. It can be visualised as service queue in which the 'front' pointer directs towards the node partition(s) to be serviced presently. 'Rear', is where the pointer to the subsystem affected by the computation of the current node is pushed in. Between the front and rear pointers the subsystems remain lined up in the queue to get serviced in the 'first come first served' basis.

Need for efficient utilisation of memory space has prompted the use of a circular queue in place of a linear queue. As more and more nodes/elements get serviced the front pointer in Fig. 2.6 move down, leaving behind a space

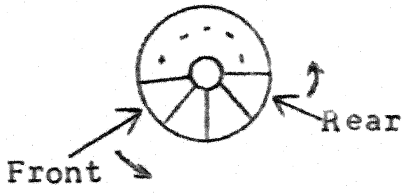


Fig. 2.6: The circular scheduler queue

that can be used (shown dotted in Fig. 2.6). So when the rear pointer reaches the fictitious end (end of the linear array simulating the queue) the queue is wrapped round to

store further elements which line up to get serviced in the space left behind by the progressing front pointer. If under this circumstance the rear pointer tends to overtake the front pointer the queue is recognised to be full and circuit simulation is terminated.

The circuit subsystems are visited in the order directed by the scheduler and evaluated. This is called a circuit trace. The trace comes to an end when there is no element left to be serviced in the queue (i.e. when the queue is empty with the front pointer tending to overtake rear pointer).

2.7 SIMULATION STEP:

The entire time domain analysis is broken up into a number of simulation steps upon discretisation. In each step all the starting nodes are scanned and updated. If any such node is found varying it is followed up by a trace directed by the scheduler along the signal flow paths. Coming to the end of trace (indicated by Q empty) next starting node is taken up. When all the starting nodes have been monitored and corresponding traces performed, the analysis is advanced to the next time step.

The time step of simulation h, is kept as an user adjustable parameter, chiefly determined by the node capacitance

value. Due to the constraint put on h in Sec. 2.4 and Sec. 2.2.1, h must be chosen small enough to, keep C_{ii}/h ratio sufficiently large. The best value of time step found for MOSIMR and kept as default value is 0.1 nsec. when the node capacitances are of the usual order of 0.1 pF.

The main aim in developing MOSIMR is improvement of speed of analysis with minimal storage requirement. However if accuracy is jeopardised in the process, one way to trade accuracy with speed etc. will be adjusting h to an even smaller value.

2.8 INITIALISATION:

Solution scheme in MOSIMR being noniterative the analysis cannot start with a guessed initial state. Rather an accurate initial state calculation is done based on a preliminary logic simulation of the digital circuit.

The circuit is assumed to be initially at steady state and the initial state computation involves evaluation of all internal and external node voltages, device currents and power dissipation under quescent condition.

2.9 DEVICE MODELS USED IN MOSIMR:

As pointed out in Sec. 2.5 in order to compute a node all the elements included in the subsystem (i.e. elements which fan-in at the concerned node) have to be evaluated for

the current time step. MOSIMR uses equational model for the devices constituting the elements. This avoids the immense space requirement of tabular models [10] but tends to increase computation time. In MOSIMR however very simple three terminal model (Fig. 2.7) is used to represent an NMOS transistor.

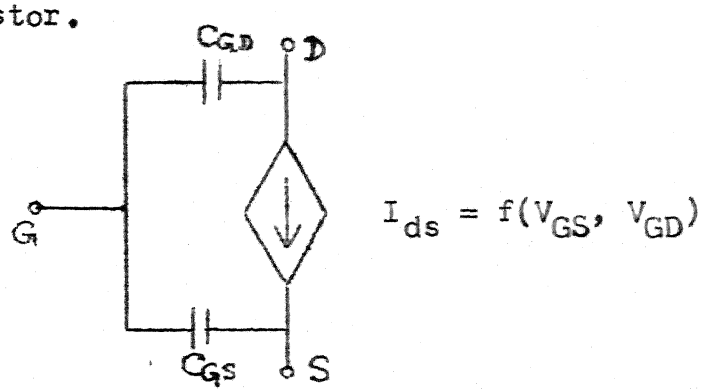


Fig. 2.7: The simple model used in MOSIMR to represent an NMOS device.

The controlled current source I_{ds} (drain current of the device) is modeled by the simple equations

$$\text{for } V_{GS} \leq V_T \; ; \; I_{ds} = 0 \qquad (2.16a)$$

$$\text{for } V_{GS} > V_T \; ; \; I_{ds} = \frac{\mu \epsilon_{ox}}{t_{ox}} \frac{\omega}{L}[(V_{GS}-V_T)V_{DS} - V_{DS}^2/2] \qquad (2.16b)$$

when $V_{DS} < V_{GS} - V_T$

and

$$I_{ds} = \frac{1}{2} \frac{\mu \epsilon_{ox}}{t_{ox}} \frac{\omega}{L} (V_{GS} - V_T)^2$$

when $V_{DS} \geq V_{GS} - V_T \qquad (2.16c)$

where, μ = surface mobility of electrons

ϵ_{ox} = dielectric constant of oxide

t_{ox} = thickness of oxide

ω = width of channel

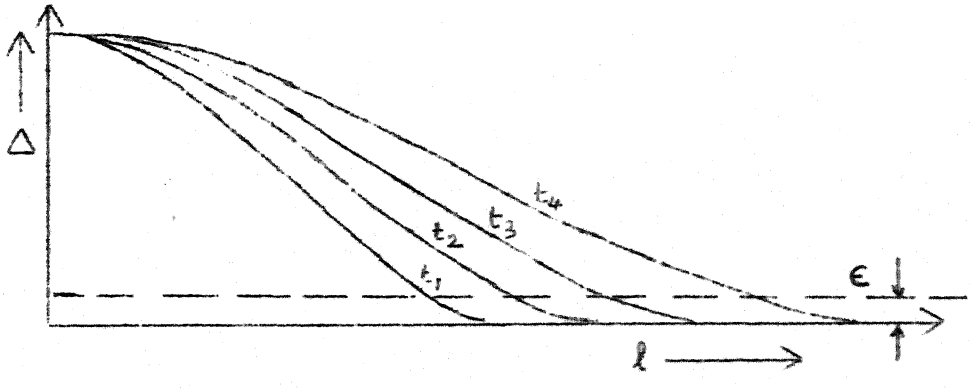
L = length of channel

V_T = threshold voltage

The simple model helps to cut down computation time, during evaluation of an element. Back gate bias effect and channel length modulation effect have not been modeled. These however do not affect the result too adversely.

2.10 SIGNAL-WAVE PROFILE DETECTION IN MOSIMR:

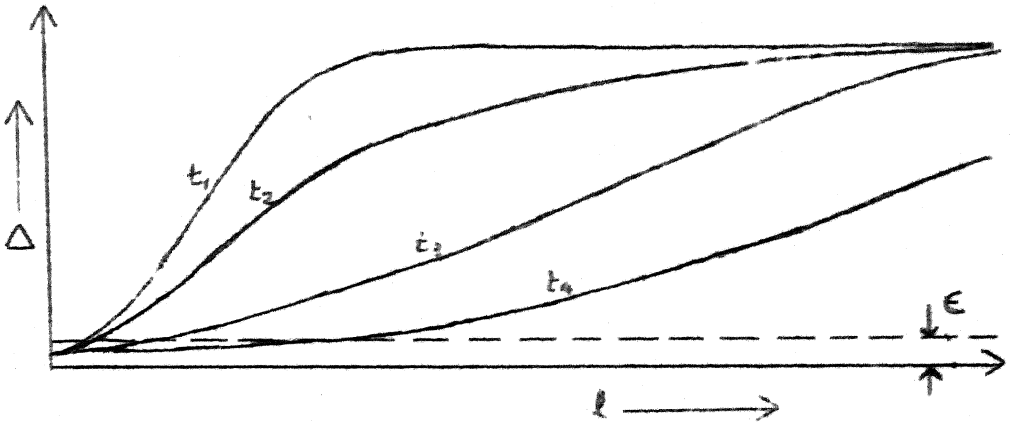
2.10.1 Signal flow Patterns in Large Circuits:



Signal-wave profile monotomically decreasing

Case - when a steady input starts to change uniformly.

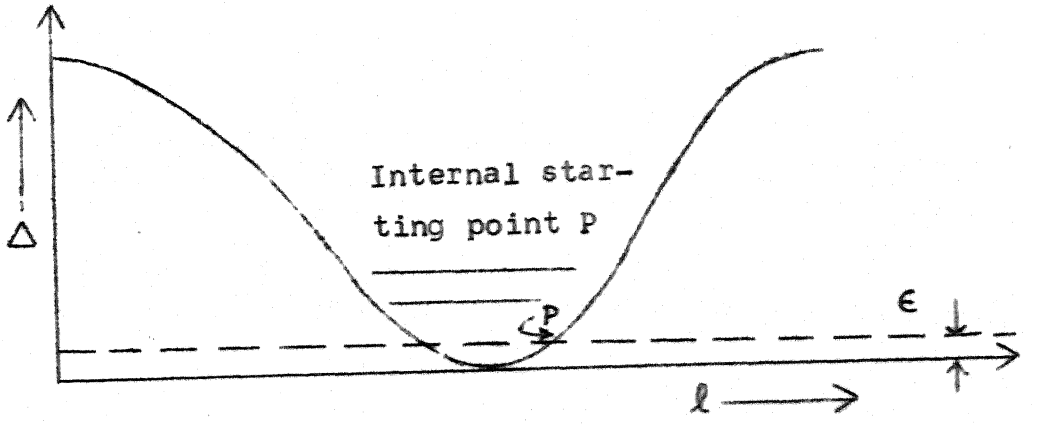
Fig. 2.8a



Signal - wave profile monotonically increasing

Case - when a varying input ceases to change and becomes steady.

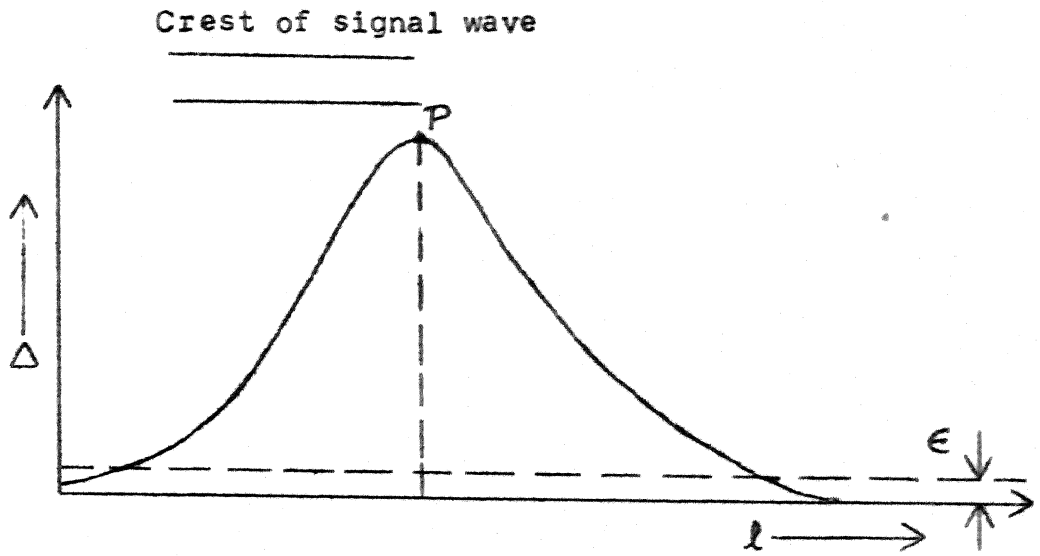
Fig. 2.8b



First monotonically decreasing then monotonically increasing

Case - when an input first varying ceases to change and then starts to vary again, sending a trough P travelling through the length of the circuit.

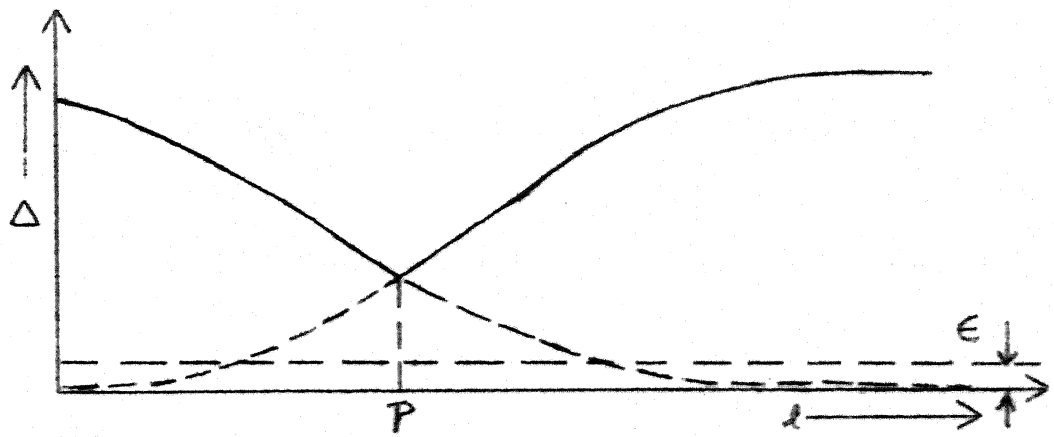
Fig. 2.8c



Signal-wave profile first monotonically increasing then monotonically decreasing.

Case - when a steady input starts varying and then again ceases to change, sending a crest P traveling through the circuit.

Fig. 2.8d



Two signal-wave profiles originated by two inputs overlapping in the common portion of signal flow path.

Fig. 2.8e

2.10.2 Latency on a Signal Flow Path:

Like majority of physical systems MOS/LSI circuits demonstrate a low pass characteristic by taking a finite time to respond to any perturbation. Perturbations in the form of signals (i.e. varying voltage levels) travel with a speed dependent on propagation delay of devices in MOS/LSI circuits. This amounts to saying that even a node that falls on the signal flow path, as found from the link-graph, (Sec. A.3.9.5) may remain latent in a particular time step if the corresponding signal wave profile does not encompass that node. This may happen in either of the two cases - i) The wave has already crossed the node, else ii) the wave is still to arrive at the node. In any case the active nodes on the signal flow path are those encompassed by the wave profile. In addition to selective trace MOSIMR is capable of finding in any time step the start and end points of the propagating signal-wave profile and hence can detect the active nodes on the signal flow path which are to be computed. Thus within a selected path again MOSIMR analyses only those nodes which are active and makes a fuller exploitation of latency.

2.10.3 Start and End Point Detection of a Signal Wave Profile:

Figs. 2.8a - 2.8e show the typical signal-wave profiles where Δ = change in voltage, l = length of the circuit on signal

flow path . Evidently l=0 is the point of application of input. The actual algorithm followed for dynamic detection of beginning and end points of a wave profile can be found in Sec. 3.5.3 and it involves a node constancy/variation flag table IFL which at the end of a time step contains the information whether a node N has remained constant

, IFL(N) = 0 , or has changed,

IFL(N) = 1 .

A change in node voltage level is recognised only when $\Delta \geq \epsilon$

where ϵ = tolerance (Fig. 2.8a)

In addition to time step h, ϵ is another variable parameter in MOSIMR which can be adjusted by the user to trade accuracy with speed. ϵ = .01 mV gives accuracy comparable to a full-fledged circuit simulator like SPICE.

The wave characteristic of signal flow will be prominent in very large circuits such as MOS/LSI. MOSIMR which is designed to take advantage of this feature and make fuller exploitation of latency, thus becomes very suitable for the simulation of large NMOS LSI's.

CHAPTER 3

PROGRAM IMPLEMENTATION DETAILS

In this chapter each aspect of macromodeling, circuit description, subcircuit expansion, linked list structuring, partitioning, ordering, logic and voltage level initialisations, timing analysis and outputting will be taken up, revealing how these have been implemented in the program MOSIMR.

The program, written in FORTRAN 10 (the language supported by the Fortran compiler of DEC 10) is of length approximately 3200 lines. It has been tested extensively on DEC 1090 with different NMOS digital circuit structures of small and medium size.

3.1 SKELETAL STRUCTURE OF THE PROGRAM:

The program is divided into 6 functional modules as follows:

1. Circuit description
2. Initialisation
3. Input
4. Analysis
5. Output
6. Control

Table 3.1 gives the names of main and accompanying subprograms and their respective functions. The following sections take up the program modules one by one and elaborate their implementation with the help of flow charts.

3.2 CIRCUIT DESCRIPTION MODULE:

The circuit description module is chiefly concerned with reading-in the linked subcircuit and linked main circuit descriptions (Fig. 3.2), expansion of subcircuits and their linking with the main circuit environment (Fig. 3.3 and Fig. 3.4) and node partitioning by fan-in/fan-out table construction (Fig. 3.5).

3.2.1 Reading-in Circuit Descriptions:

The circuit description by the user is in terms of circuit macromodels, defined as 'elements' in MOSIMR, in a fixed format specified in user's manual (Appendix B). An element can be a logic gate (e.g. inverter, two-input NAND and NOR gates etc.), a functional block (e.g. a latch) or simply a pass transistor. New functional blocks can be defined by the user as subcircuits.

In main circuit description an element is specified by its name, number, input/output nodes, the name and number of the next brother element at each input node and the name(s)

and number(s) of the successor element(s) at the output node(s) (exact format given in Appendix B). Specifications for an element of type K and number N are stored in Nth row of the table meant for K, as element record. Any main circuit element having a link with a subcircuit at node N is marked correspondingly with an 'L' in the specification. Name and number of such an element are kept stored in row N of LINK table for future reference during linking of subcircuit with main circuit.

The subcircuit description is analogous to that of main circuit. Only here all node and element numberings are local, and global linkages with the main circuit are missing. All these therefore need processing during subcircuit expansion before final element records can be obtained. Hence the circuit description module upon reading-in the subcircuit description, stores specifications of each element, along with its name and number, in each row of work area (i.e. IWKAR Table), temporarily. The pointers to the zone in IWKAR (i.e. the IWKAR address where the description starts and the IWKAR address where the description ends), where the description of a subcircuit is recorded, along with the corresponding subcircuit name, are stored in a row of the pointer-table IPTR.

3.2.2 Subcircuit Expansion and Global Linking:

While reading the input data when a subcircuit call is encountered, the IPTR table is searched with the subcircuit name as key, to obtain the pointers to the zone in IMKAR where the subcircuit description can be found. Then element records are fatched from the zone, processed and finally stored in appropriate element tables, sequentially,

The processing involves

1) offsetting local element nos. with the help of MEL (refer Table 3.2).
2) offsetting the local internal node nos. with the help of MAXNOD (refer Table 3.2)
3) Replacing dummy external nodes of the subcircuit with corresponding actual nodes provided by the user.
4) Setting up links with the main circuit environment with the help of LINK table.

Processing 1-3 are elaborated in Fig. 3.3. As the subcircuit is expanded and new elements and nodes add up to main circuit, the element count vector MEL and node count variable MAXNOD are updated.

Processing 4, which involves linking of the subcircuit with the main circuit, can be of three types.

The subcircuit element fetched from temporary store in IWKAR and the main circuit element kept recorded in row N of LINK table are to be linked at node N. The possibilities are:

i) The main circuit element becomes the successor (Succ.) of the fetched elem.

ii) The fetched element becomes the succ. of the main circuit element.

In this case any future linking at node N would be with the presently fetched element. Hence the present entry in row N of LINK Table has to be replaced with the name and number of currently fetched subcircuit element.

iii) The fetched and main circuit element can be related as brothers at the common input node N.

The direction of next brother linking is decided as follows -

a) When the signal flows from the main circuit to the subcircuit, the fetched element is appended as next - brother (NB) at the end of the NB chain of the main circuit element in LINK table.

b) Otherwise the main circuit element is appended as NB at the end of the NB chain of the fetched element.

Since in this latter case the fetched element becomes the leading element (Sec. A.3.6) of the overall NB-chain at node N, any future linking at N will have to refer to it. Hence present entry in row N of LINK table is replaced by the currently fetched subcircuit element.

All the possibilities (i) to (iii) are envisaged in Fig. 3.4 and record of each linked and processed subcircuit element is finally stored in appropriate element tables.

The end result is a linked list structure stored in the element tables, with each element record holding a pointer (corresponding to each input/output node) to its NB/succ. element.

3.2.3 Node Partitioning:

The node partitioning algorithm by fan-in table construction has been elaborated in Fig. 3.5a, 3.5b. The wired logic connection which involves joining of two or more gate outputs is not supported by MOSIMR. In fact tying up of two or more drain/source nodes is treated as a conflict (Figs. 3.1a, 3.1c, 3.1d, 3.1e) and is inadmissible except in cases shown in Figs.3.1c-3.1e involving pass transistors. Here node N has,as opposed to ordinary case (Fig. 3.1b), more than one element controlling (Sec. A.3.9) it. Such

multiple controllability over N is resolved by building a partitioned subsystem that consists of all the elements that fan-in at N. (Sec.A.3.11.1). To compute N it is necessary to evaluate all the elements in the partition assigned to node N.

In MOSIMR the work space IWKAR which becomes free after the expansion of all subcircuits, is used to store the fan-in as well as fan-out table for each of the conflicting nodes. As the only allowable conflict in MOSIMR is associated with D/S node(s) of pass transistors the node partitioning algorithm in the circuit description module takes up each pass transistor and explores the source node 'N' for a possible conflict. IFAN (N,3) stores the number of predecessors, as obtained from circuit description, at node N(refer Table 3.2). If at the explored source node the number of predecessors is > 1 a conflict is found to exist. The pulled end (considered as the drain terminal) of the pass transistor as shown in Fig. 3.1c is always in conflict. The multiple controls (Sec. A.3.9) over these conflicting nodes are then tabulated in the 6th and 7th column of IWKAR by storing the names and numbers of the elements fanning-in (Sec. A.3.11.1) at N (the node of conflict). Along side, the fan-out elements

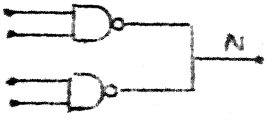


Fig. 3.1a: The inadmissible wired connection of gate outputs.

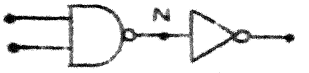


Fig. 3.1b: Ordinary case with no conflict.

Fig. 3.1c: Gate output connected to D/S terminal of pass trans.

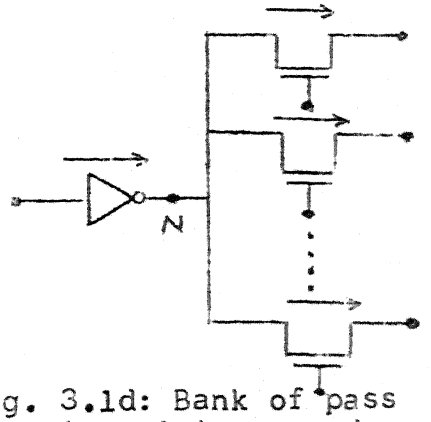


Fig. 3.1d: Bank of pass transistors used as demultiplexer

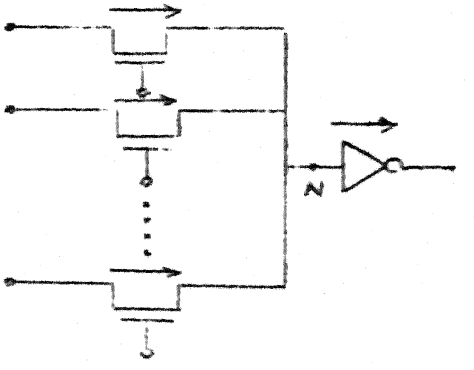


Fig. 3.1e: Bank of pass transistors used as multiplexer.

(Sec. A.3.11.2) at N are also stored in the 8th and 9th columns of work area. Finally the pointer to fan-in /fan-out table location in the work area for a particular node N is stored in LINK (N,3) for future reference.

Construction of the fan-in table achieves node partitioning in the given circuit. The node plus its fan-in elements form a partitioned subsystem.

3.3 INITIALISATION MODULE:

Initialisation of the given circuit is performed in two sweeps. The first sweep carries out a full fledged logic simulation to determine the logic states of all the circuit nodes.

The second sweep takes up each element and computes its output node voltage(s) and internal node voltages and device currents and the quescent element-power dissipation based on the input logic states.

3.3.1 Logic Simulation:

The nodes are initiated with a don't know state '2'. The user is required to specify the logic state of each input and the internal bistable elements. Starting a circuit trace from each of these nodes of known logic states, elements are visited in the order directed by the linked list, and their output logic states, computed.

The first three columns of IWKAR wrapped around form a circular queue to schedule the servicing of elements (Sec. 2.6) and conduct an initialisation trace. As the

logic state of a node N is determined, the leading number (Sec. A.3.6) of the chain of elements observing that node (Sec. A.3.9), is entered at the rear of the service queue to carry on the trace to the next logic level. On the other hand if the gate output remains undetermined after servicing or is found to be already determined, further trace is terminated by not making any entry to the service Q. Naturally to determine the ouput logic state of the element currently being serviced (in that, the front of the queue or any of its next brothers) either one or many of its inputs should have a dominant logic state (i.e. 1 for NOR gates, 0 for NAND gates), or else logic states of all the input nodes should be known. The initial logic simulation has been detailed in Figs. 3.6, 3.7.

Finally some floating nodes may remain (such as output node of an off pass trans.) which are now visited and the user is asked to specify initial logic states for these, to his liking. As each floating node is specified, a trace is conducted from that node to initialise the subsequent levels (if these are undetermined). Fig. 3.8 gives the details of floating node initialisation scheme.

3.3.2 Node Voltage and Device Current Initialisation:

At the end of logic simulation the user is asked to specify a number of NMOS device parameters, interactively

through the TTY. These include threshold voltage (V_T), surface mobility of electrons (μ), thickness of oxide (t_{ox}), dielectric constant of the oxide (ϵ_{ox}), W/L ratio of load and driver transistor etc. Any parameter that remains unspecified by the user is assigned a default value by the program. With the help of these parameters intrinsic transconductance β is calculated for the enhancement mode driver device, the depletion mode load device and the pass transistor.

Finally based on the known input logic states the simple drain current equations (2.16) are solved to get device currents, internal and output node voltages, and power dissipation in the element under quiescent condition. Following assumptions are made during solution of the initial element state:

i) Input logic state — Corresponding voltage level.

1	$V_{DD} = 5V$
3	$V_{DD} - V_T = 4V$
0	$< V_T$

ii) Circuit is initially at steady state, implying zero current flow through 'on' pass trans. and node capacitances.

iii) The saturation current CLS (computed from the device parameters) flows through the load transistor under steady state when the gate is 'on'. The drain currents are zero when the gate is 'off'.

iv) An 'off' pass transistor has zero subthreshold leakage-current.

The actual voltage level for logic low (assumed < V_T) has no influence on restoring logic elements. However in a steering logic element like a pass transistor, the drain and source voltages track each other if the element is 'on'. If the input voltage, in such a case, is not known during evaluation of the pass transistor, the output voltage remains unevaluated. This may happen if the input logic is '0' and its voltage is not yet computed. MOSIMR assumes a default 0.5V for the output, in such cases. But this approximation causes some error in the voltage initialisation (Sec. 5.2).

3.4 INPUT MODULE:

The inputs can be either externally defined (EXDEF) or internally generated (INGEN). EXDEF inputs can be any arbitrary waveforms, values for which are tabulated for every time step over the entire simulation period. This table of externally defined inputs is provided by the user as input data to the program.

An INGEN input on the other hand is a regular input such as i) a continuous 1 phase/2 phase non-overlapping clock waveform. ii) A rising/falling edge iii) A positive/negative pulse or simply a iv) constant logic low/high level. These are inputs useful in digital circuits and are internally generated by MOSIMR based on user given specifications. The specifications include i) Rise/fall time ii) High/low period of a pulse etc. and are accepted from the user in the input module as illustrated in Fig. 3.10.

3.5 ANALYSIS MODULE:

After computation of the initial state the linked and partitioned circuit is subjected to the specified input signals and its time domain response is analysed. The analysis module which follows after the input module does this task by conducting a selective trace along the signal path and computing only the active node partitions.

3.5.1 Selective Trace:

The same circular service queue implemented on IWKAR and utilised by initialisation trace, becomes the mainstay of analysis trace as well. An analysis trace is initiated by a change in the external input which perturb the system that is at steady state to start with. In each simulation step the external inputs are scanned and updated. Any change in

any input at node N greater than a preset limit ϵ, is recognised as a signal which will affect the elements observing (Sec. A.3.9) N. These are the elements obtained as succ. (or its next brothers) at N or as an entry in the fan-out table for node N. The leading element (Sec. A.3.6) of these is entered at the rear of the scheduler queue. This way the partitioned subsystems are ordered on the basis of observation (sec. A.3.9) by the scheduler, as each node is computed. Figs. 3.11 - 3.13 show the scheduling algorithm in detail.

3.5.2 Computation of a Node Partition:

'Front' of the scheduler queue points at the element being currently serviced.

Servicing an element actually involves finding the contribution by the element in the present time step to the voltage(s) at its output node(s), N. The contribution is computed by solving the eqn. (2.14). Using current values of the decoupling inputs (Sec. 2.3.2.1) (recognise this as the Gauss Seidel method), each element in the partitioned subsystem is evaluated (which involves updating of its device currents, internal node voltages and power dissipation stored in the real element table) and its contribution to node N, found out. Accumulating the contributions by all

elements which have control (Sec. A.3.9) over N, the node is updated for the current time step. For a node of conflict (Sec. 3.2.3) the elements included in the subsystem, are listed against the node in fan-in table. Figs. 3.14 and 3.15 give the details of node voltage updating in a partitioned subsystem with individual element evaluation.

3.5.3 Signal Wave Profile Detection:

This has been elaborated in Figs. 3.12, 3.13. In essence it is based upon the following points:

i) NCON is a flag storing the information whether the currently computed node was constant in the previous time step.

IFL (NODE) is the constancy flag of the currently computed node in the present time step.

IFL(N) is the constancy flag of the starting node N from where the analysis trace has originated.

ii) Let VOUT be the evaluated value in this time step and VNODE (NODE) the previous time step value of the voltage at node 'NODE'. Then,

$$\Delta = \text{ABS(VOUT - VNODE (NODE))}$$

is the change in voltage at NODE in this time step.

If $\Delta \geq \epsilon$ recognise it as a signal at NODE

$\Delta < \epsilon$ recognise it as no signal at NODE

Based on this simple premise the analysis trace detects the starting and finishing points of a traveling signal wave in each time step. While conducting the analysis trace, upon evaluating each node on the signal flow path, Δ is checked against ϵ, a preset limit.

If $\Delta \geq \epsilon$ the flag IFL(N) is checked and if it shows that the starting point of the current analysis trace is constant, in the present time step, at once the current node 'NODE' is recognised as the new starting point of the signal wave. Further progress of current trace is stopped and the name and no. of the predecessor element on the trace path at 'NODE' is stored in IWKAR (NODE,10) - IWKAR (NODE,11) to mark the starting point of fresh traces in the current and subsequent time steps. This way the analysis of leading part of a signal flow path, which is found inactive, can be skipped (refer Fig. 2.8b).

In a similar fashion end of the signal wave can be detected when $\Delta < \epsilon$ (with starting node 'N' of the trace not constant) and rest of the trace, beyond the signal wave profile, being redundant, is abandonned (refer Fig. 2.8a).

This simple scheme of signal wave profile detection, incorporated in MOSIMR, is useful in exploiting latency of very large circuits, where only a fraction of nodes/elements

remain encompassed by the propagating signal wave profile, at a given instant of time, on a signal flow path.

3.6 OUTPUT MODULE:

Before the start of timing analysis the time step and simulation period are specified by the user in nano-seconds. The time step DELTA ($0 < DELTA \leq 1$ n-sec) is the one used in analysis for discretisation (by B.E. integration), whereas the printing/plotting step is computed in the output module as shown in Fig. 3.17. It is kept roughly around 1 nsec. for all choices of DELTA.

At each printing/plotting step, the current value of the requested node voltages and element powers, obtained from VNODE and real element tables (refer Table 3.2) are printed and/or plotted against current simulation time. The user can ask for at most ten node voltages and ten element powers to be outputted this way.

Total power, if requested for, is calculated and printed/plotted similarly. The total run time required by the job is printed at the end of the output file.

3.7 CONTROL MODULE:

At the end of simulation period the control module interacts with the user, offering him the options to i) CONTINUE

ii) STOP or to iii) PRESTART further simulation. Thus user is allowed to steer the future course of the simulator (Fig. 3.17). In case of CONTINUE'ing, the user is asked to specify the new simulation period (he is also allowed to specify inputs (not all) for the new simulation period). The present state of the circuit is preserved, and further analysis is continued from this state,as control is returned to the beginning of analysis trace.

In the case of RESTART'ing, the control is returned to the beginning of Initialisation module. Present state of the circuit is destroyed and analysis starts by reinitialising the circuit, performing a logic simulation.

Table 3.1:

Main program MOSIMR uses following subprograms to implement the 6 functional modules.

Name of the subprogram	Functions performed
SUB1 SUB2 SUB3 SUB4	Circuit description module.
LINIT ITRACE LEVAL	Logic initialisation
VINIT	voltage/current initialisation
INSPEC INGEN	Input module
ATRACE NEVAL SBRTN ELEVAL	Analysis module
PLOT	Output module

N.B.: The control module is included within the main program itself.

Table 3.2: A glossary of arrays and variables used for structuring data in MOSIAR.

Name of array/ variable	Dimensions	Functions
IR	1	Read pointer. Useful in diagnostic messages to locate error in input data.
IMKAR	100 x 18	Work Area. i) Temporarily stores sub-circuit descriptions (before the subckts. are expanded) - Circuit Description module. ii) Used as a circular queue while conducting logic initialisation trace - Initialisation module. iii) Holds the specifications for the Ith input in IMKAR (I,13)- IMKAR(I,17) - Input module. iv) a) IMKAR(I,1)- IMKAR(I,3) used as a circular queue to conduct Analysis trace. b) IMKAR(I,6) - IMKAR(I,9) - holds fan-in/fan-out table for appropriate nodes. c) IMKAR (I,10) - IMKAR(I,11) stores the name and no. of the predec. elem. at node I for starting a trace from the internal starting point I.

Name of array/ variable	Dimensions	Functions
		iii) LINK(I,3) holds the pointer to the INKAR row from where onwards fan-in/fan-out table is kept stored for node I. - Circuit Description module.
IPTR	10 x 3	Pointer table. i) Pointer to the stored description of each subckt. IPTR(I,1) - stores subckt. name IPTR(I,2) - holds the INKAR address from where the description begins. IPTR(I,3) - holds the INKAR address where the subckt. description ends. No. of subckts. that can be described should be < no. of rows of IPTR i.e. < 10 - Circuit Description module.
MEL	10	Element Count. MEL(I) stores the no. of elements of type I, present in the circuit.
MAXNOD	1	Node Count. No. of global nodes present in the circuit.
IFAN	100x3	Predecessor table. Stores the predecessor element type and no., for each node . IFAN(I,1) - stores elem. type of the leading predec. element

contd...

Name of array/ variable	Dimensions	Functions
		IFAN(I,2) - Elem. no. of the leading predec. elem. IFAN(I,3) - stores the no. of predecessor elements present at node I. - Circuit Description module.
EXTERN	9 x 3	External link. This contains in each row a dummy node no. and the name and no. of the subckt. elem. which has external connection with the main ckt. at the dummy node.
EXTNOD	9	This stores the actual nodes which replace the dummy nodes of the called subckt.
VNODE	100	Node voltage vector. VNODE(I) stores the voltage of node I in the current simulation step.
CAP	100	Node capacitance vector: CAP(I) - stores the equivalent (device + stray) capacitance at node I, with respect to ground.
INVER etc.	50x6	Element table. Each row of an element table contains an element record. For example element record of the Ith inverter is as follows:

contd...

Name of array/ variable	Dimensions	Functions
		INVER(I,1) - stores the input node no. INVER(I,2) - stores the output node no. INVER(I,3) - INVER(I,4) - store the name and no. of the next brother element at the input node. INVER(I,5) - INVER(I,6) - store the name and no. of the succ. element at the output node.
REINV etc.	50 x 3	Real element table. This contains the real records of elements. For example record for the Ith inverter is stored as follows: REINV(I,1) - stores the load transistor current in the present simulation step. REINV(I,2) - stores the driver transistor current in the present simulation step. REINV(I,3) - stores the power dissipation by the inverter in the current simulation step.
IFL	100	Node constancy flag. IFL(I) = 0, when node I remains constant in the present time step.

contd..

Name of array/ variable	Dimensions	Functions
		IFL(I) = 1, when node I varies in the present time step.
IB or NB	4	Temporary storage vector. These temporary storages are used to pass successor or next brother elements into Initialisation/ Analysis trace routines during scheduling.
NSTEP and NPER	1	NSTEP - simulation step counter NPER - stores the specified period of simulation.
The following arrays and variables are used in NEVAL and ELEVAL routines which are respectively involved in evaluating a node and evaluating an element.		
NOUT	2	NOUT(1) and NOUT(2) store the two possible output nodes of the element being computed in ELEVAL routine.
NODOUT	2	NODOUT(1) and NODOUT(2) contain the two possible output nodes of an element on signal flow path. These are evaluated by the NEVAL routine.
VOUT	2	The temporary store of the new voltage at NODOUT(I), after the node has been updated by the NEVAL routine, for the current time step.

contd...

Name of array/ variable	Dimensions	Functions
IEFL	2	Evaluation flag. IEFL(I)=1 if NOUT(I) is already evaluated in the present time step. IEFL(I) = 0 if NOUT (I) is still not evaluated in the present time step.
CUMULC	2	Cumulative capacitor current. CUMULC(I) cumulatively stores the contribution of each fan-in element towards the node capacitor at output node NODOUT(I). Finally it holds the net capacitor current charging the node NODOUT(I), when all fan-in elements at NODOUT(I) have been evaluated.
DELC	2	Element contribution to capacitor current DELC(I) holds the contribution of the element, being evaluated by ELEVAL, towards the capacitor charging current at node, NODOUT(I).

contd...

Name of array/ variable	Dimensions	Functions
CUMULG	2	Cumulative output conductance. CUMULG(I) cumulatively stores the contribution of each evaluated fan-in element towards the output conductance at node NODOUT(I).
DELG	2	Element contribution to the output conductance. DELG(I) holds the contribution of the element being evaluated by ELEVAL, towards the conductance at node NODOUT(I).

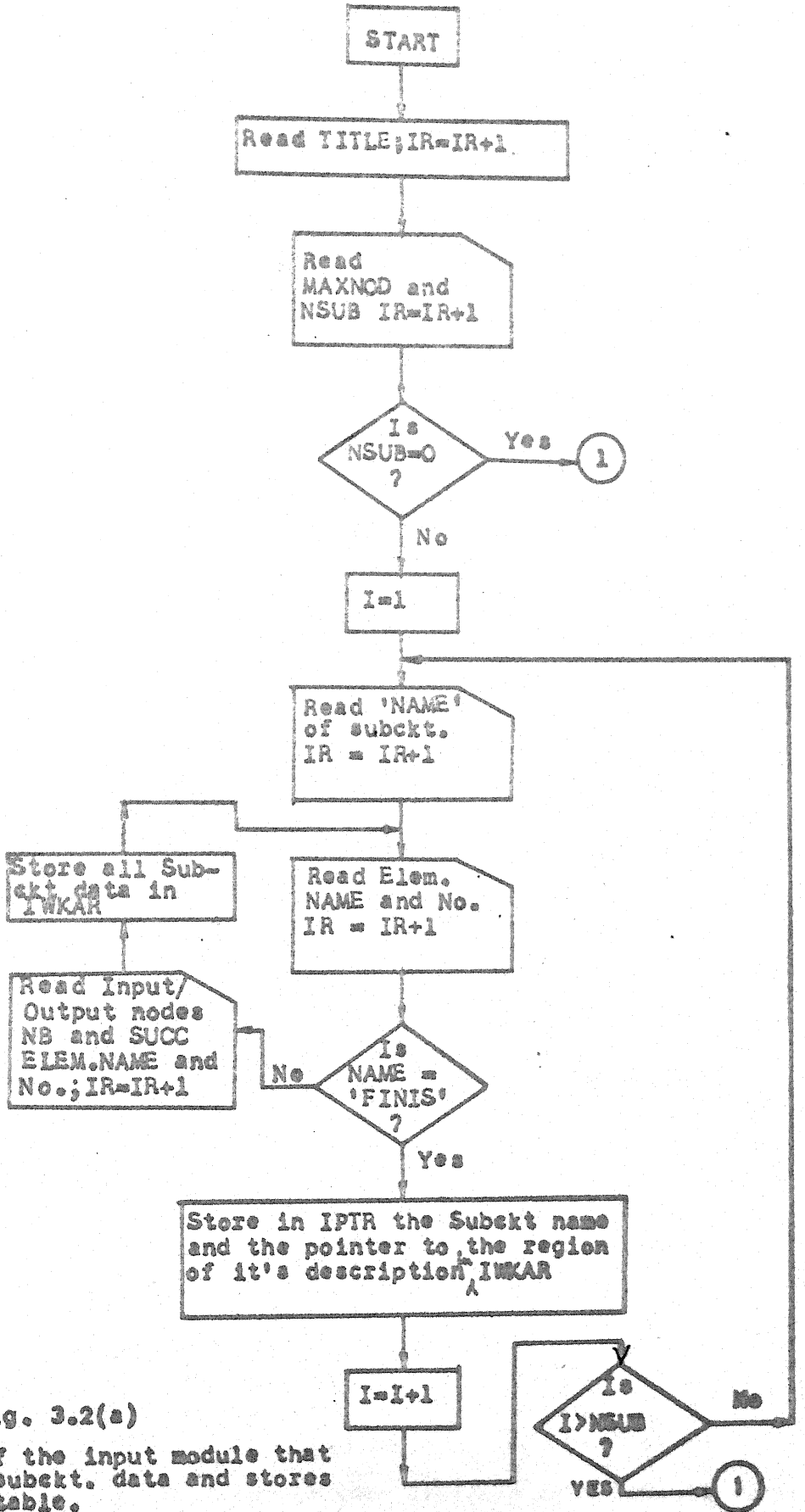


Fig. 3.2(a)

Segment of the input module that reads-in subckt. data and stores in IWKAR table.

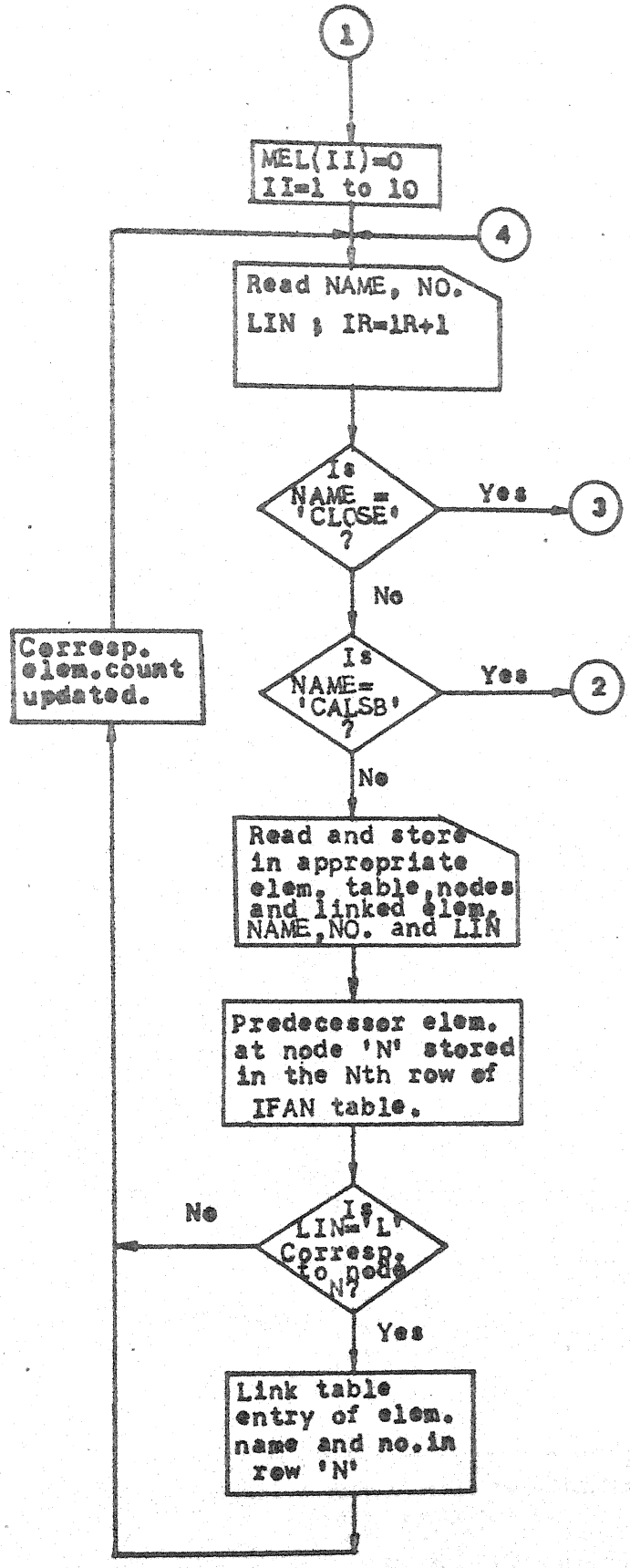


Fig. 3.2(b)

Segment of the input module that reads-in main circuit description and stores in appropriate elem./input tables as elem./input records.

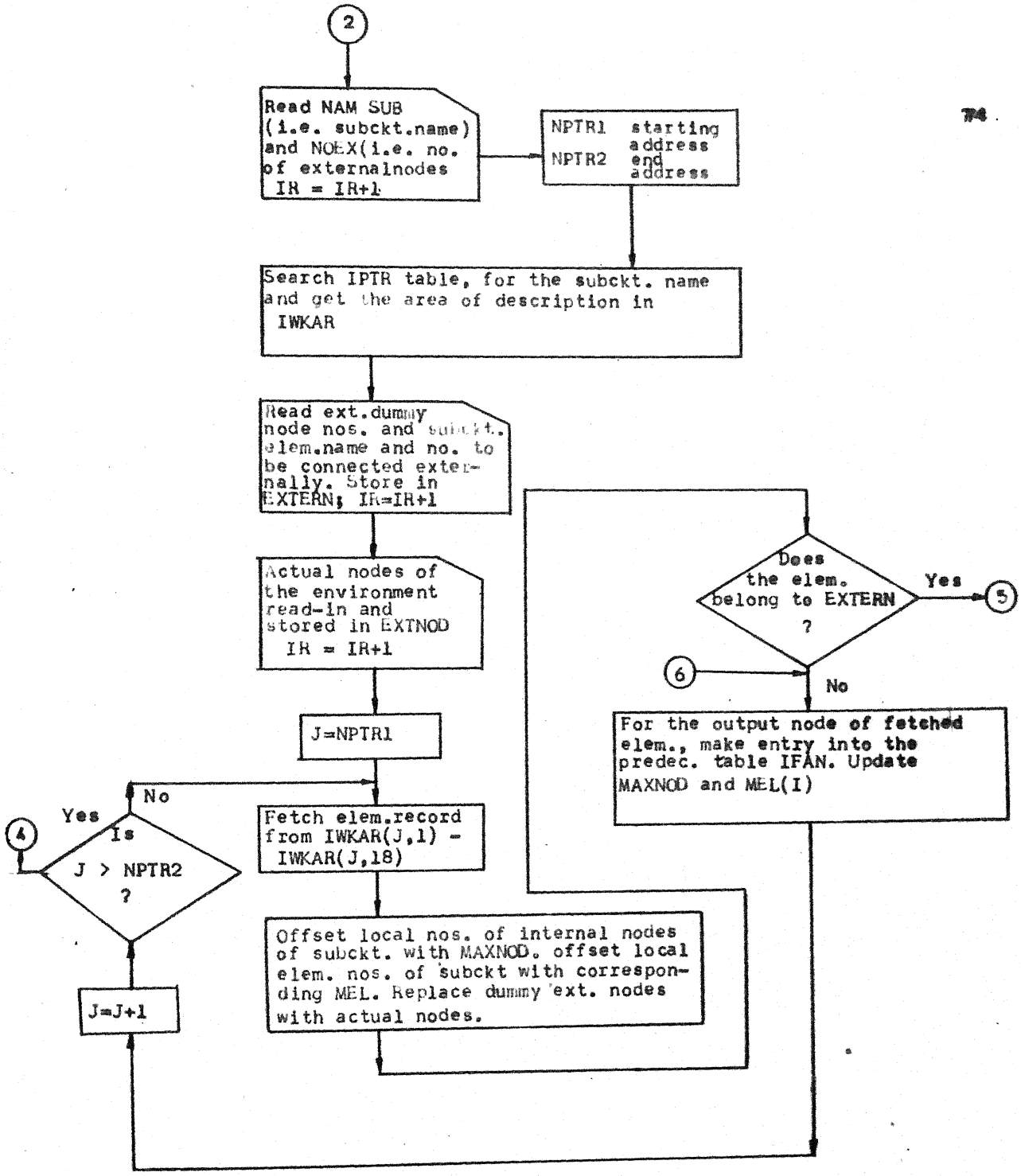
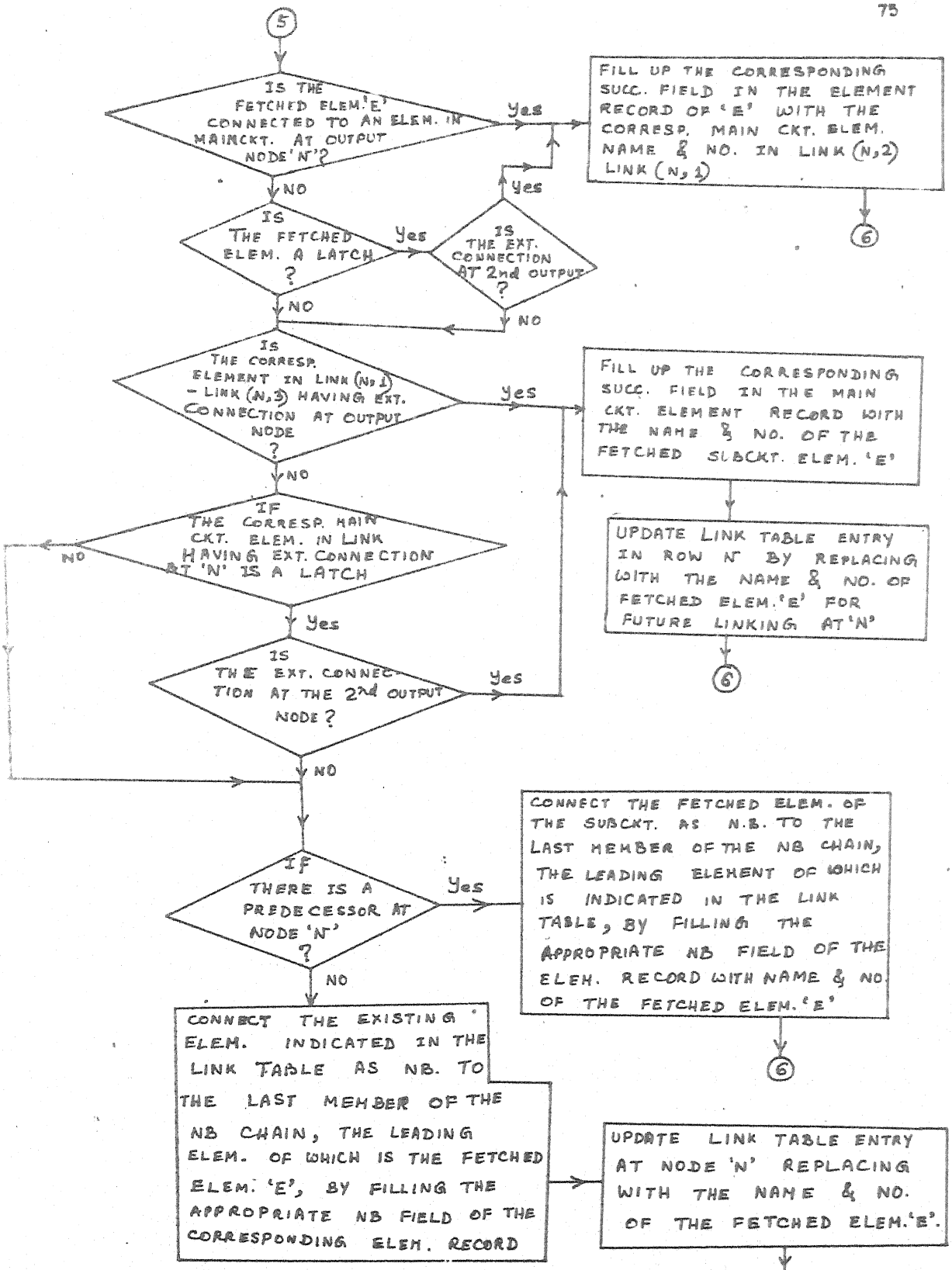
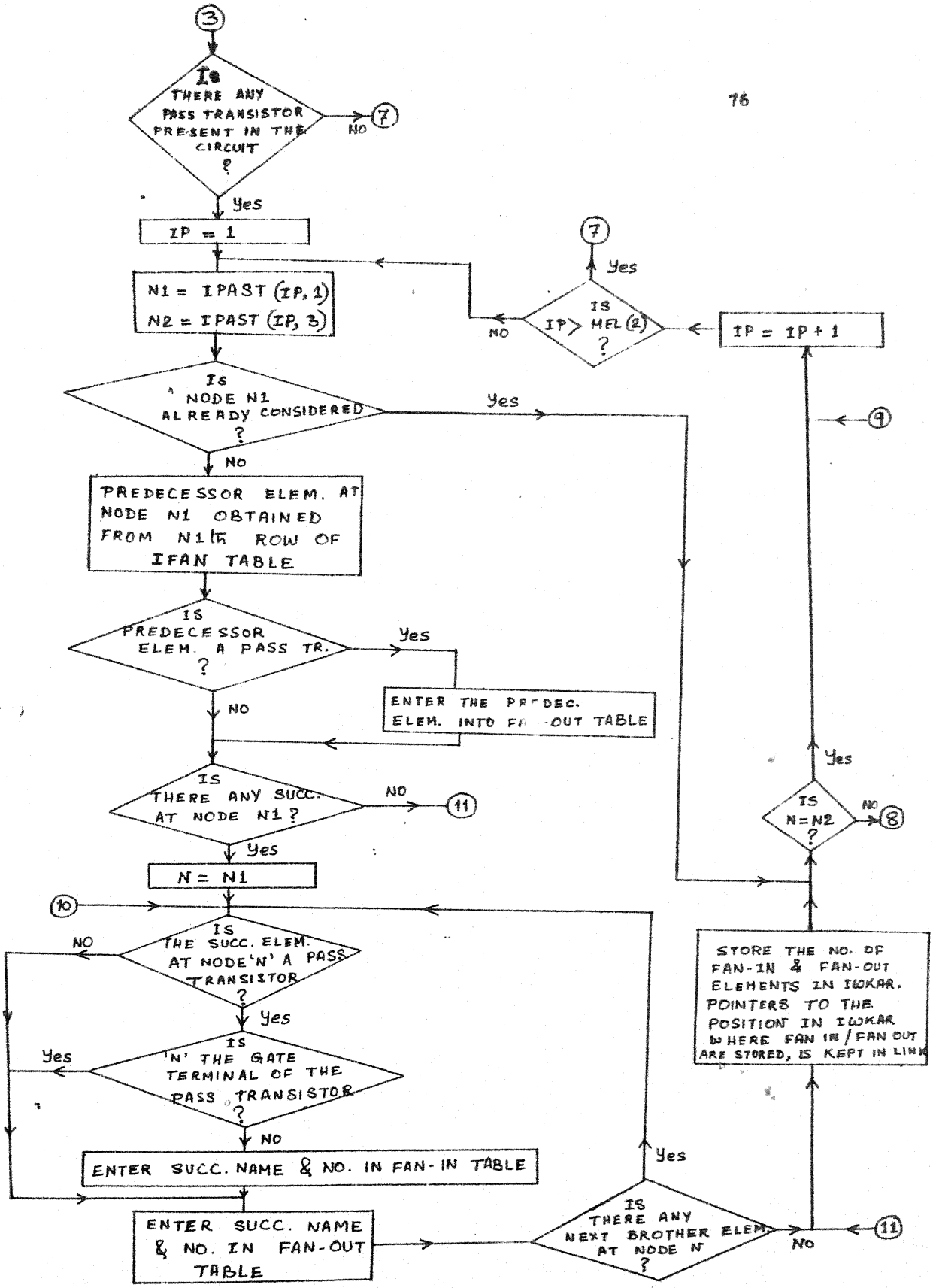


Fig. 3.3: Catering a subckt. call by expanding a local subckt. in global main ckt. environment.



LINKING OF SUBCKT. WITH MAIN CKT. ENVIRONMENT

Fig. 3.4



FAN-IN / FAN-OUT TABLE CONSTRUCTION FOR NODE N1

FIG. 3.5 a .

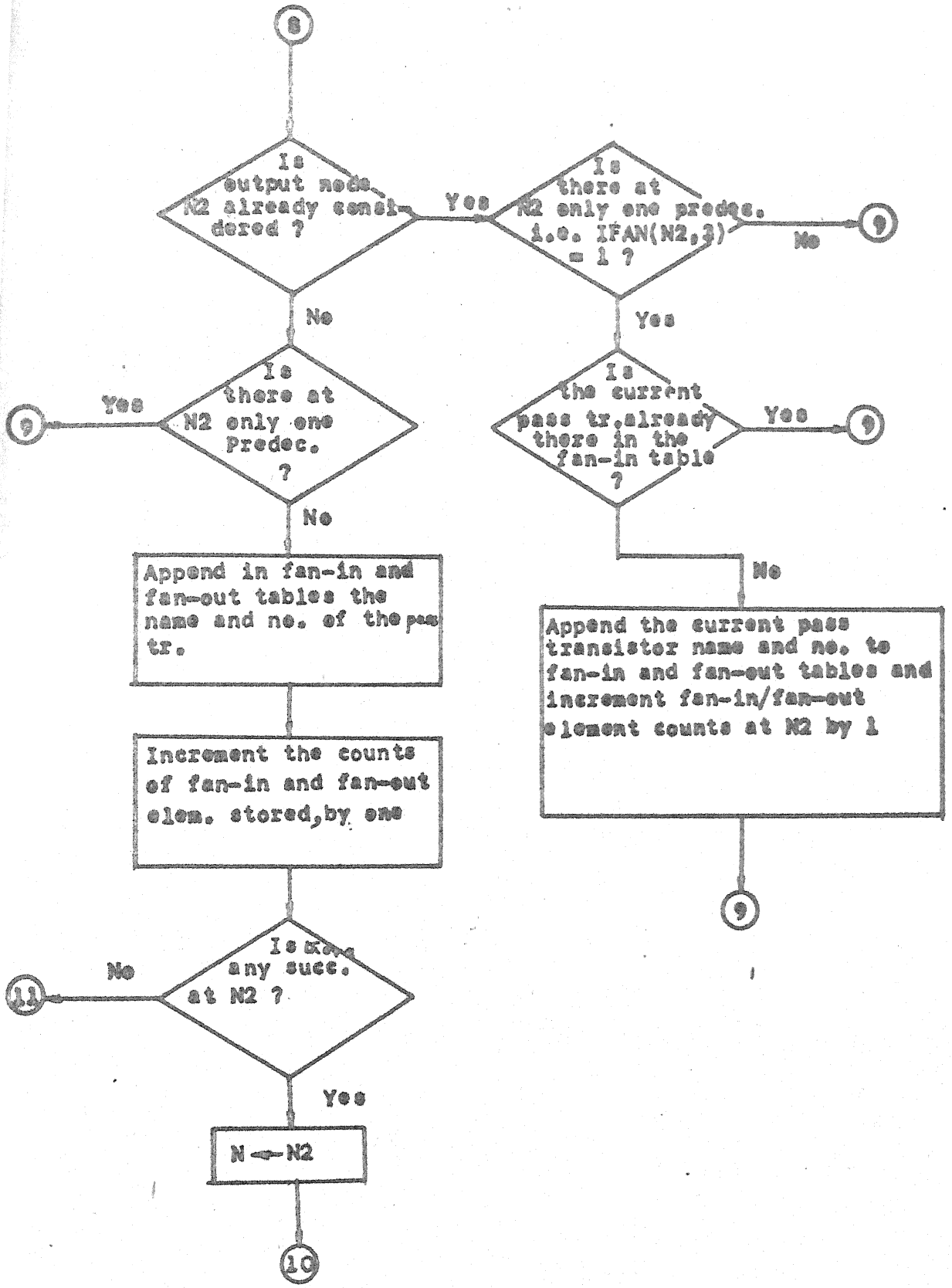


Fig. 3.5b: Fan-in/fan-out table for node N2 (contd..)

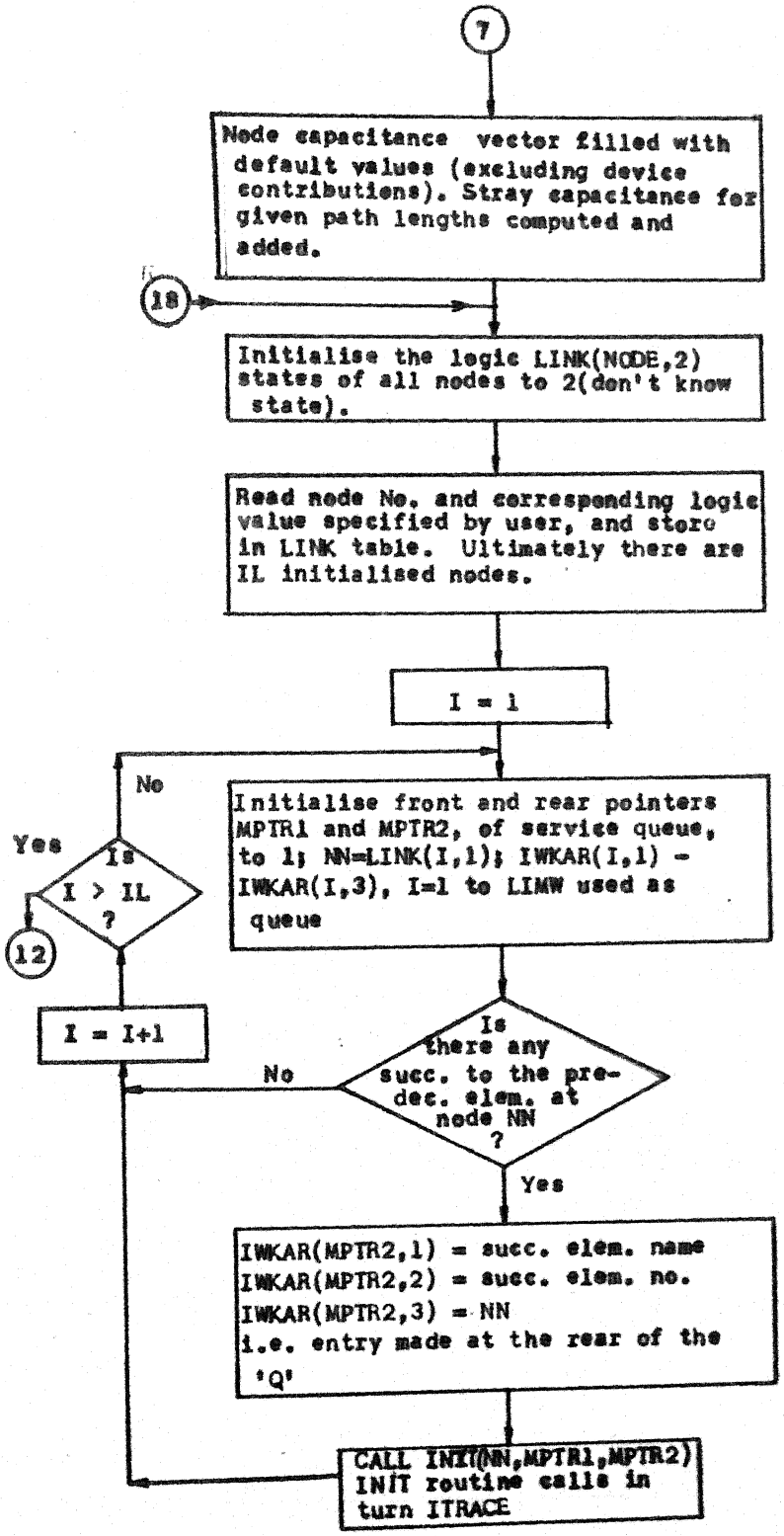


Fig. 3.6: Initialisation of the logic analysis trace

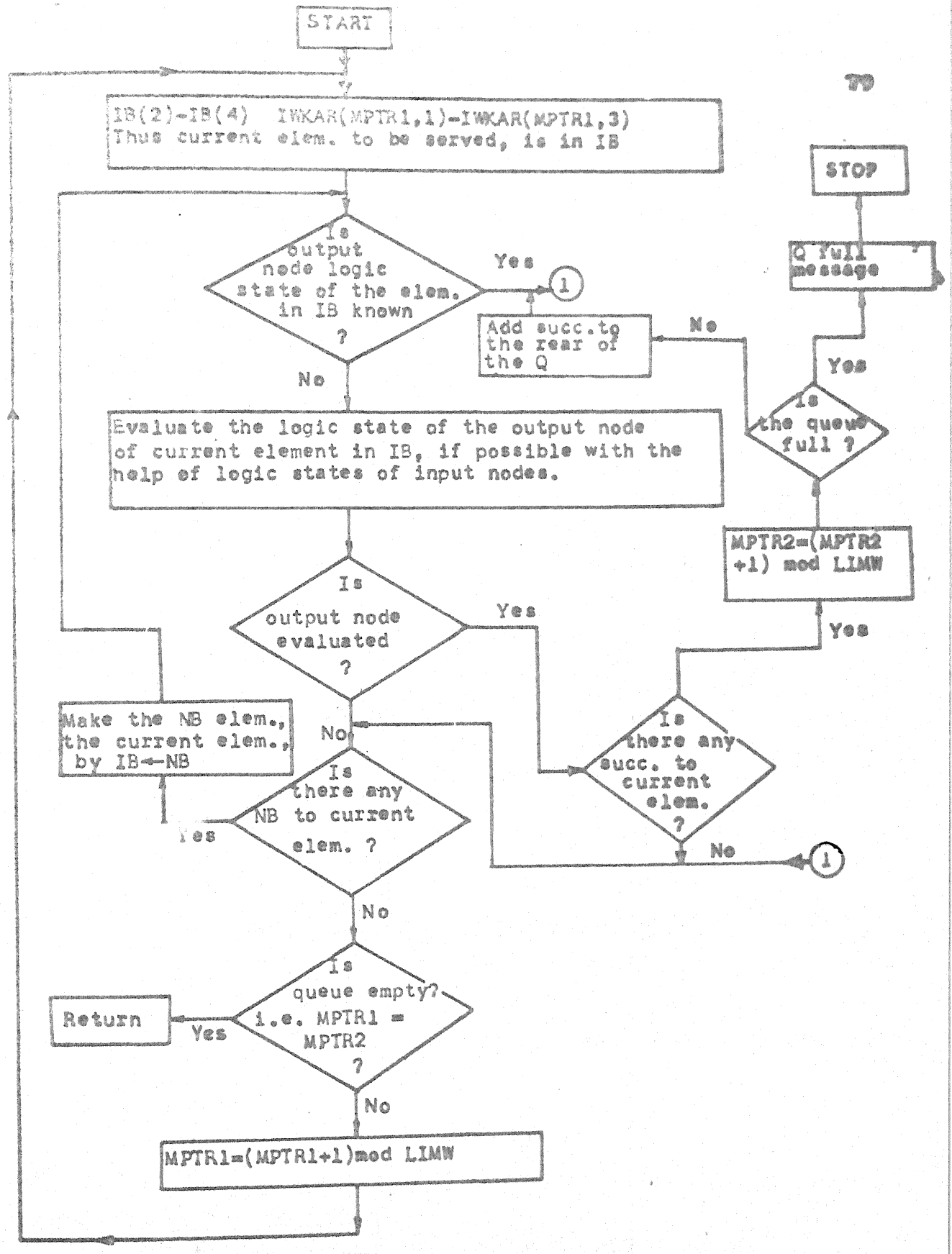


Fig. 3.7: Logic analysis trace conducted by the routine ITRACE.

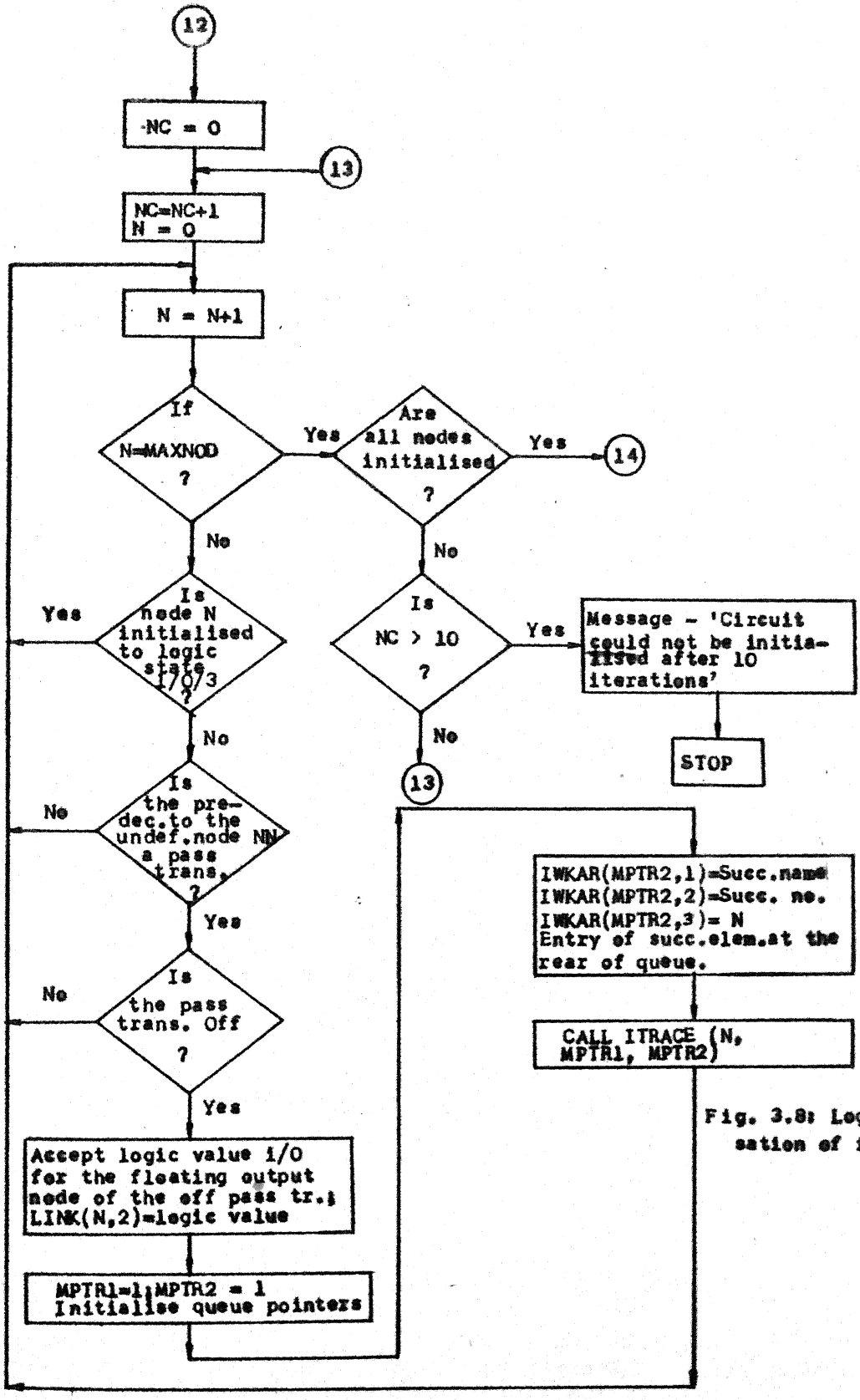


Fig. 3.8: Logic Initialisation of floating nodes

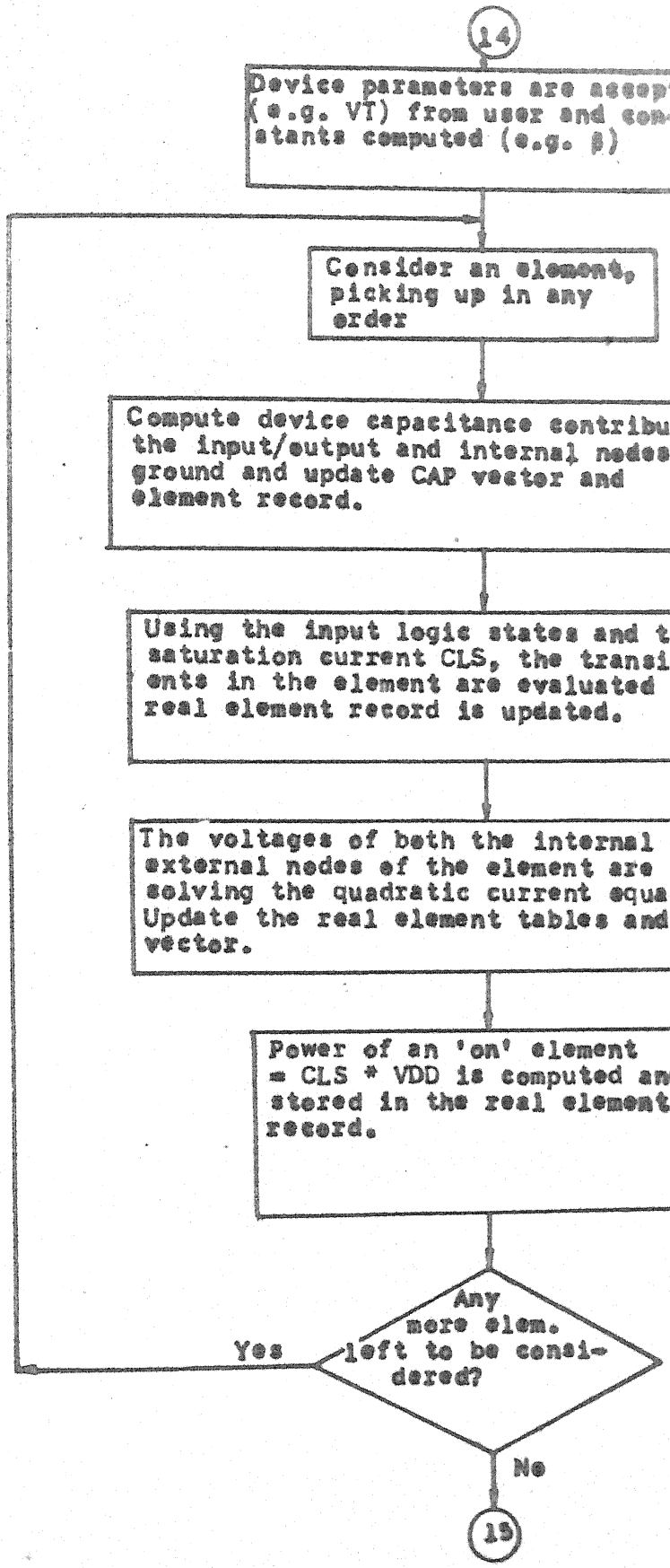
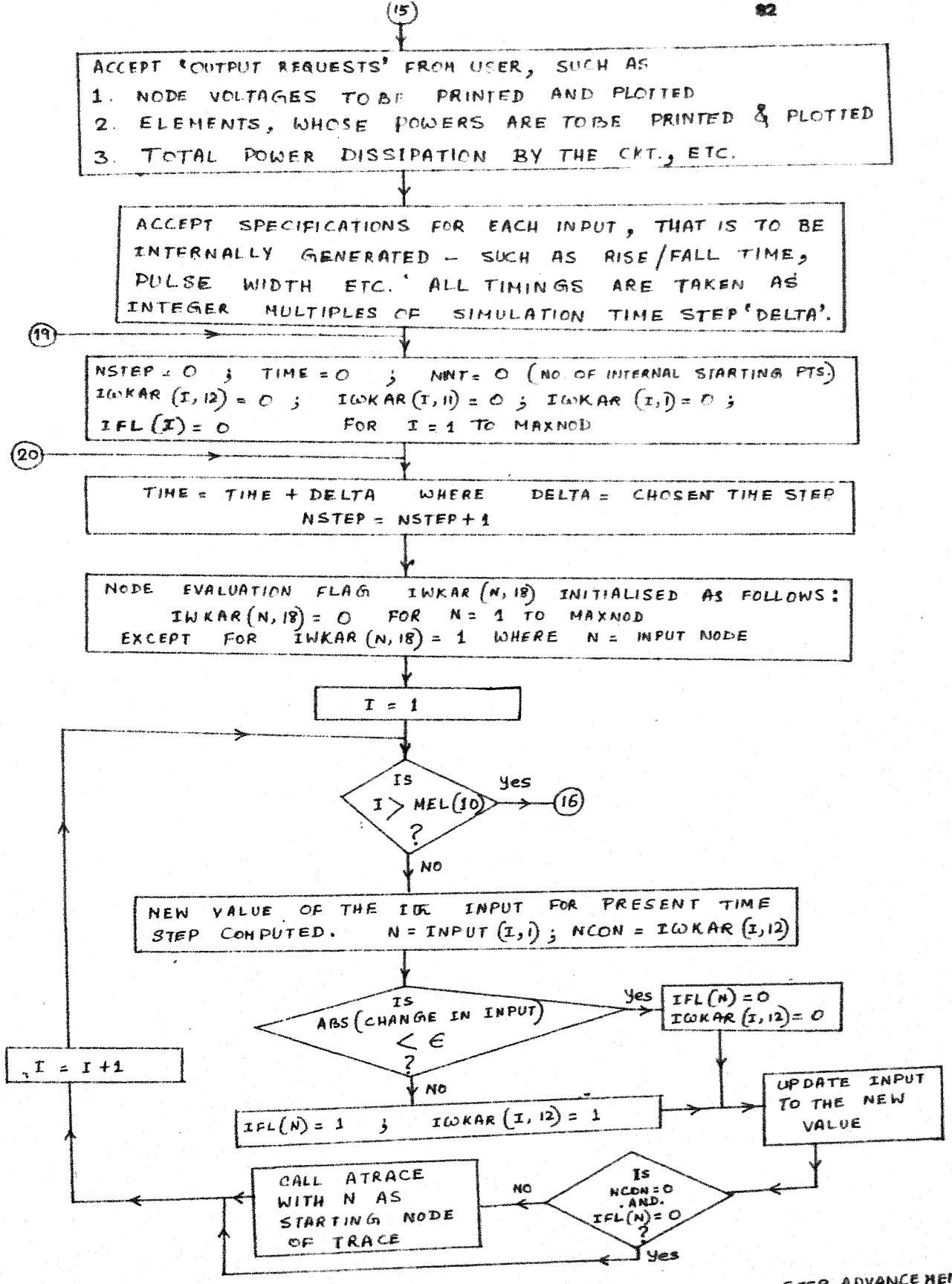


Fig. 3.9: Node voltage and device current initialisation and computation of capacitance contribution by gates at input/output nodes.



OUTPUT REQUESTS, SIGNAL INPUT SPECIFICATIONS, SIMULATION TIME STEP ADVANCEMENT AND INPUT GENERATION FOR PRESENT TIME STEP

FIG. 3.10

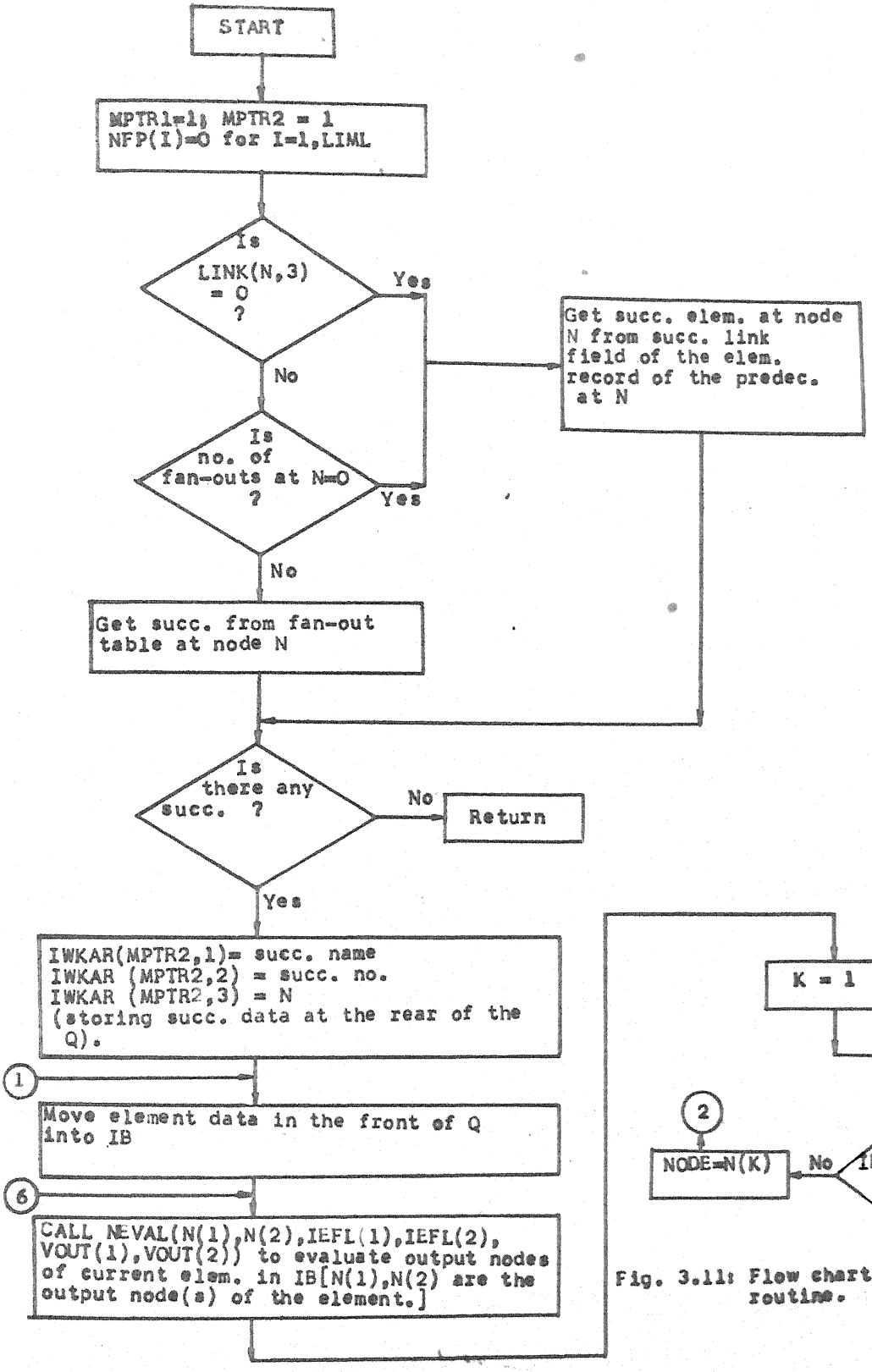
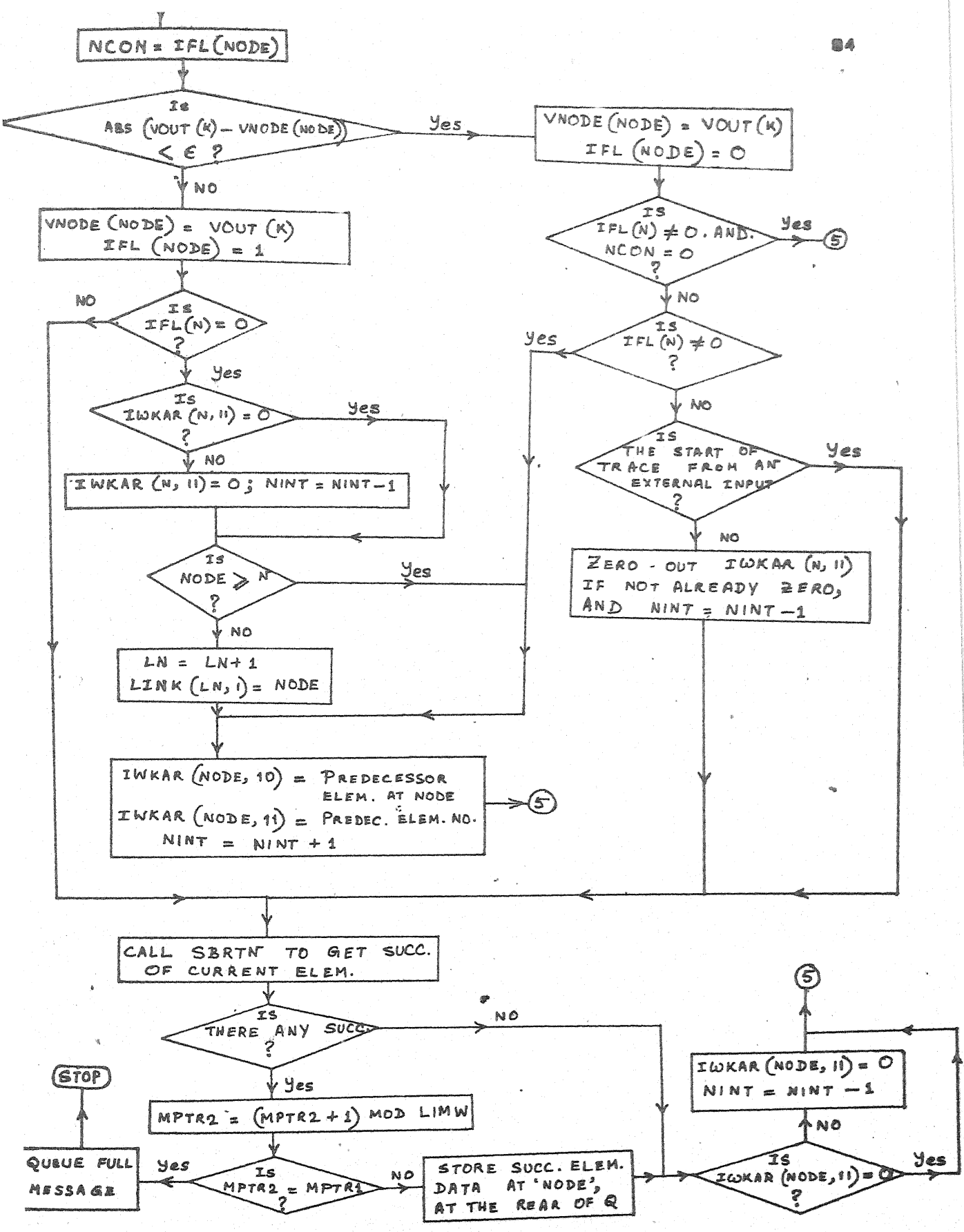


Fig. 3.11: Flow chart of analysis trace routine.



SEGMENT OF ATRACE ROUTINE INVOLVED IN DETECTION OF SIGNAL WAVE PROFILE FIG. 3.12

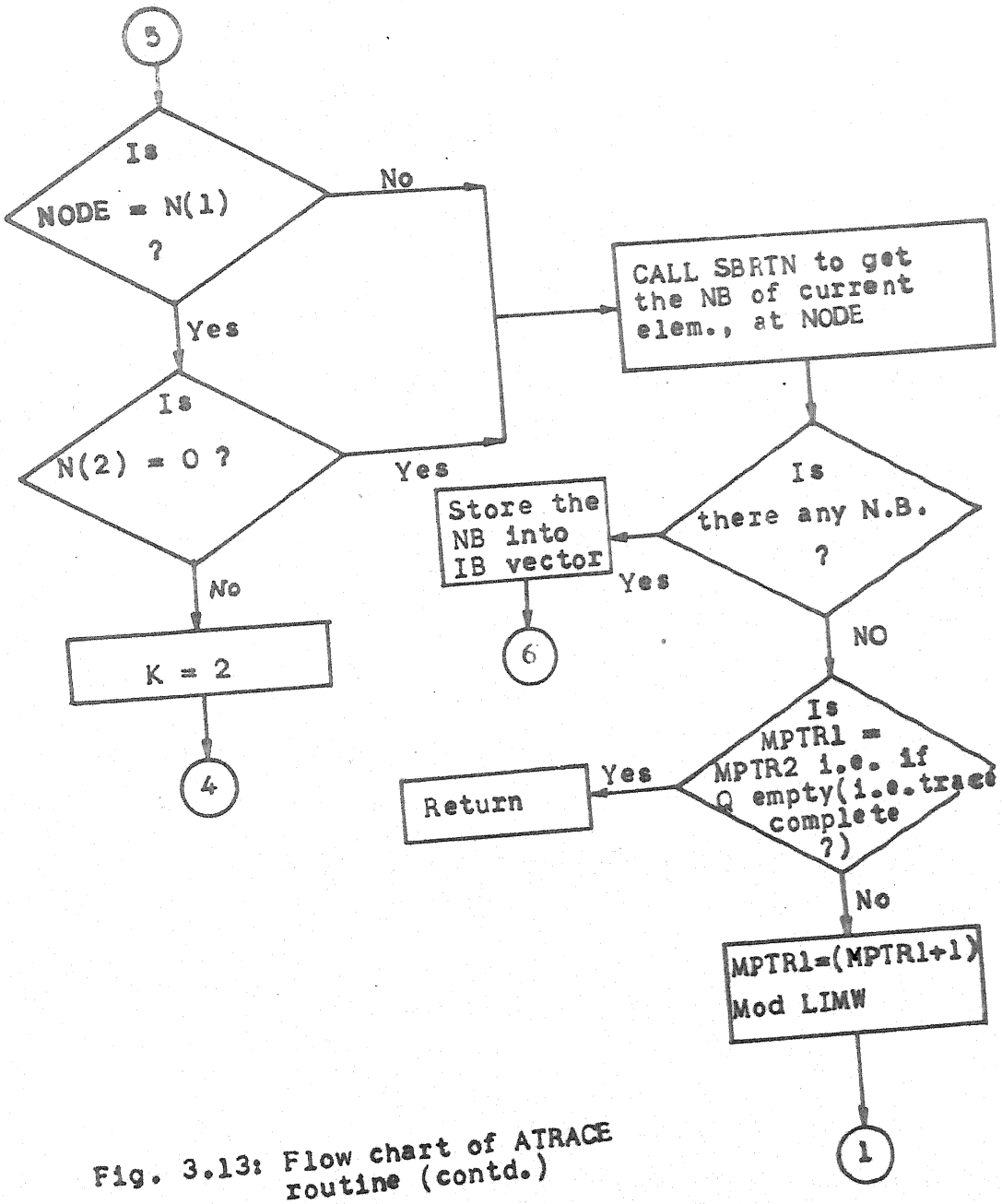
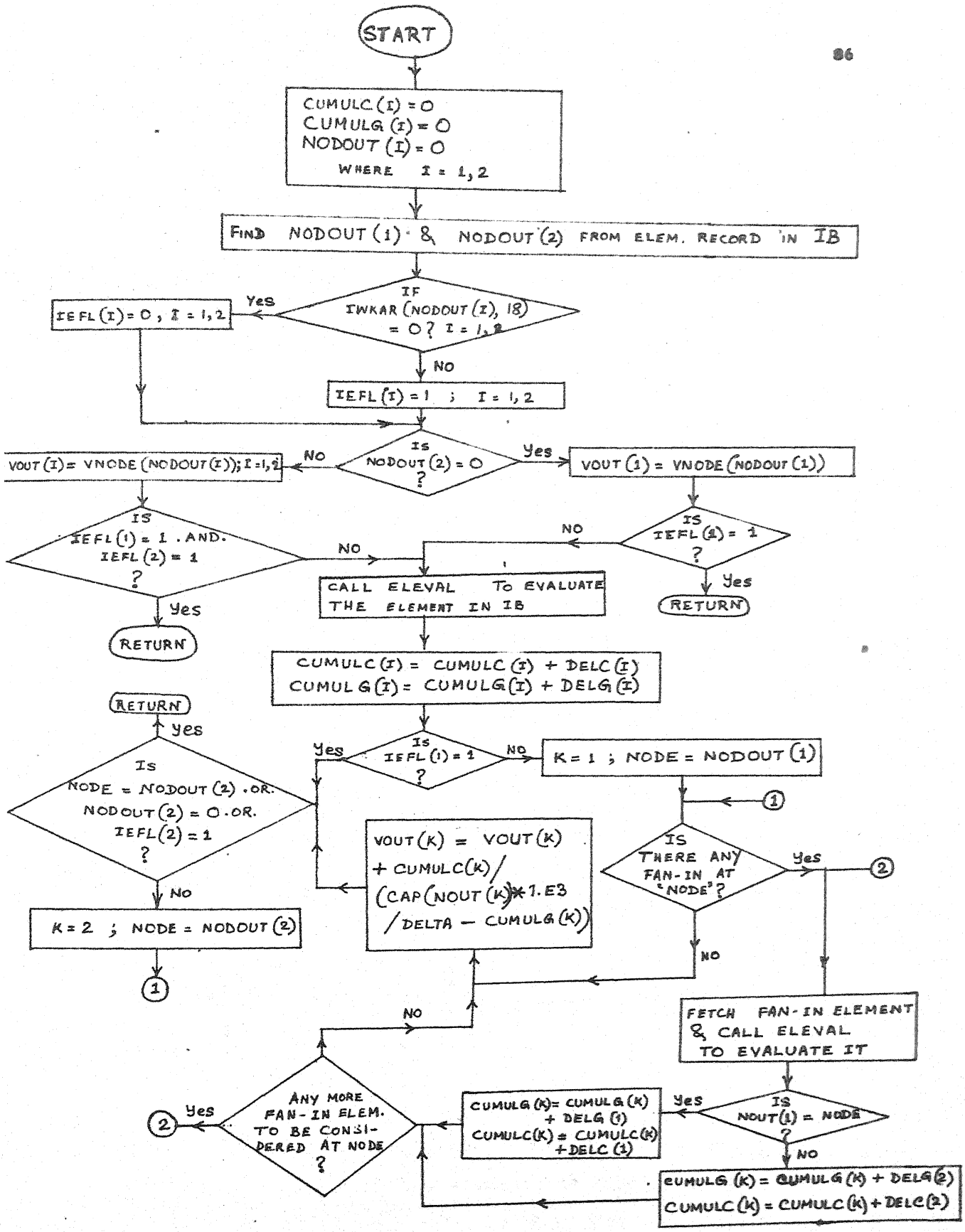


Fig. 3.13: Flow chart of ATRACE routine (contd.)



FLOW CHART OF NEVAL ROUTINE

FIG. 3.14

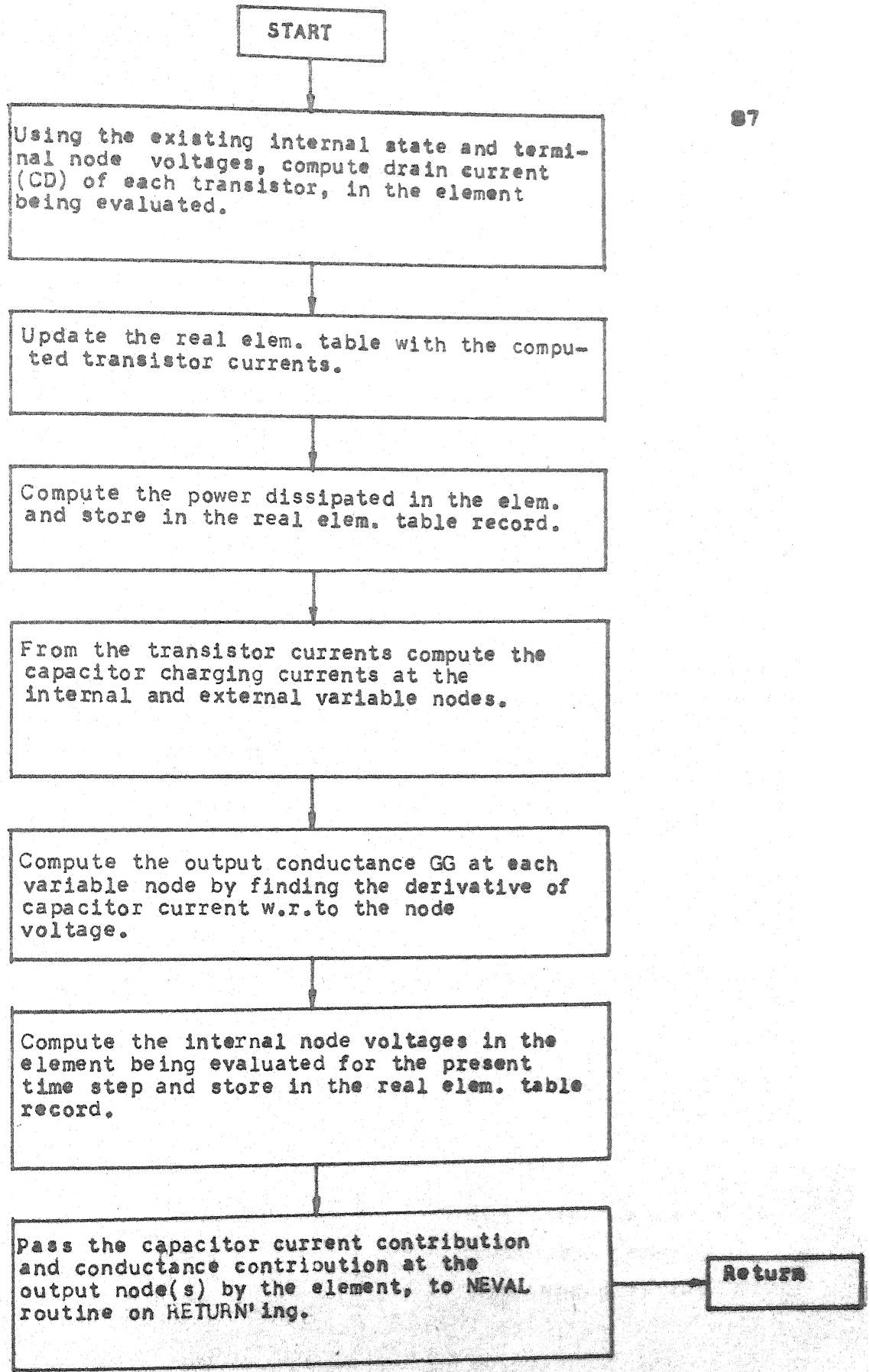
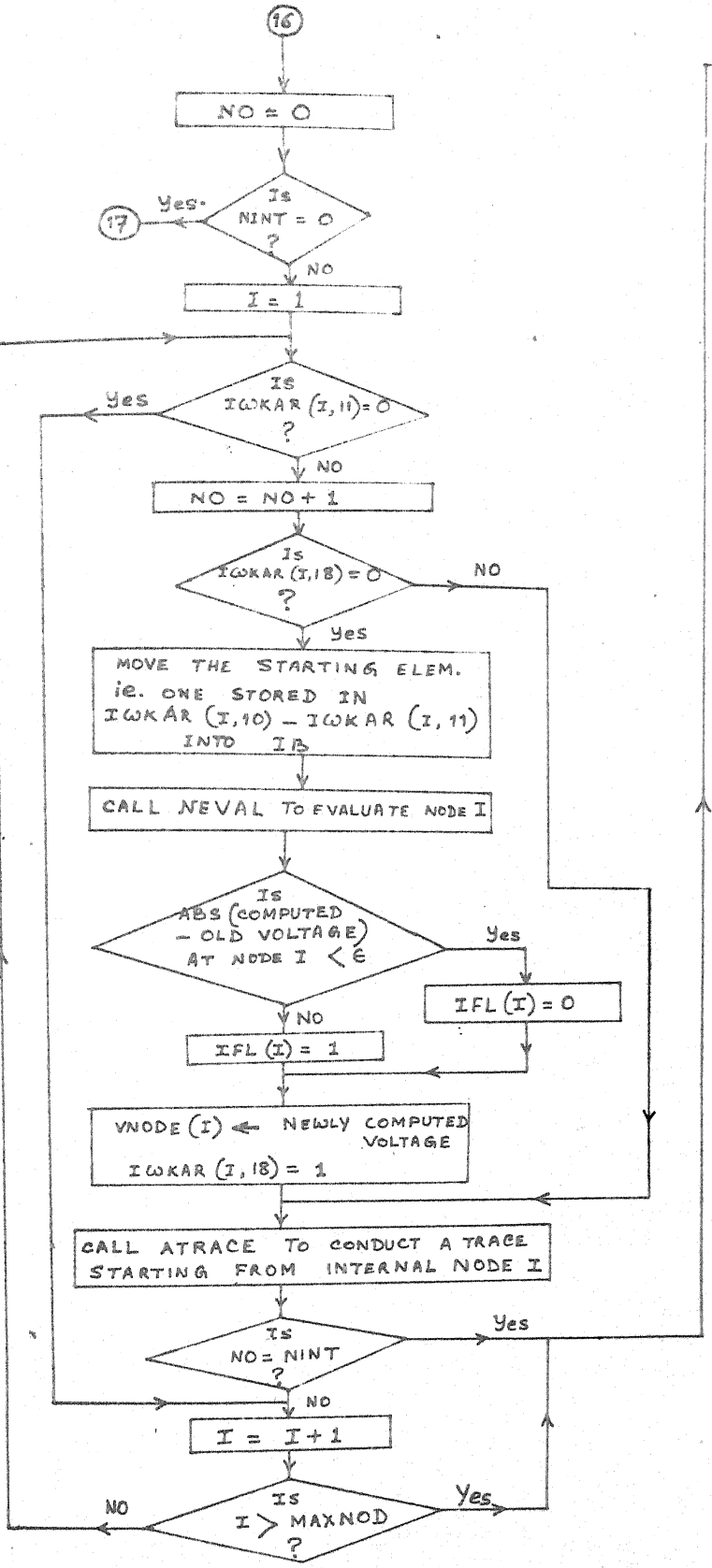
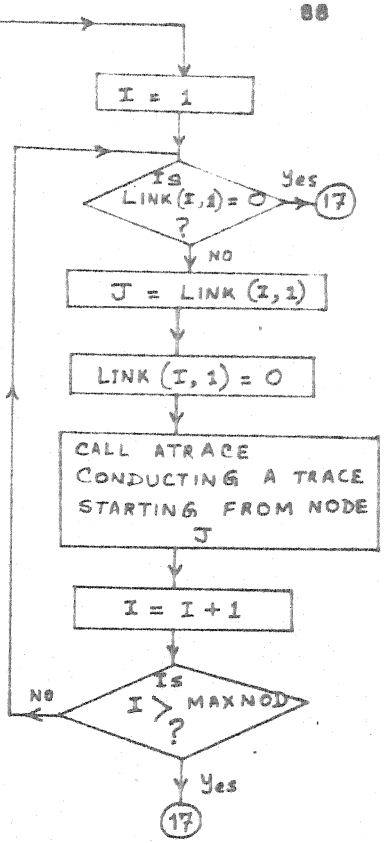


Fig. 3.15: Flow chart of ELEVAL routine





ANALYSIS TRACE CONDUCTED FROM AN INTERNAL STARTING NODE

FIG. 3.16

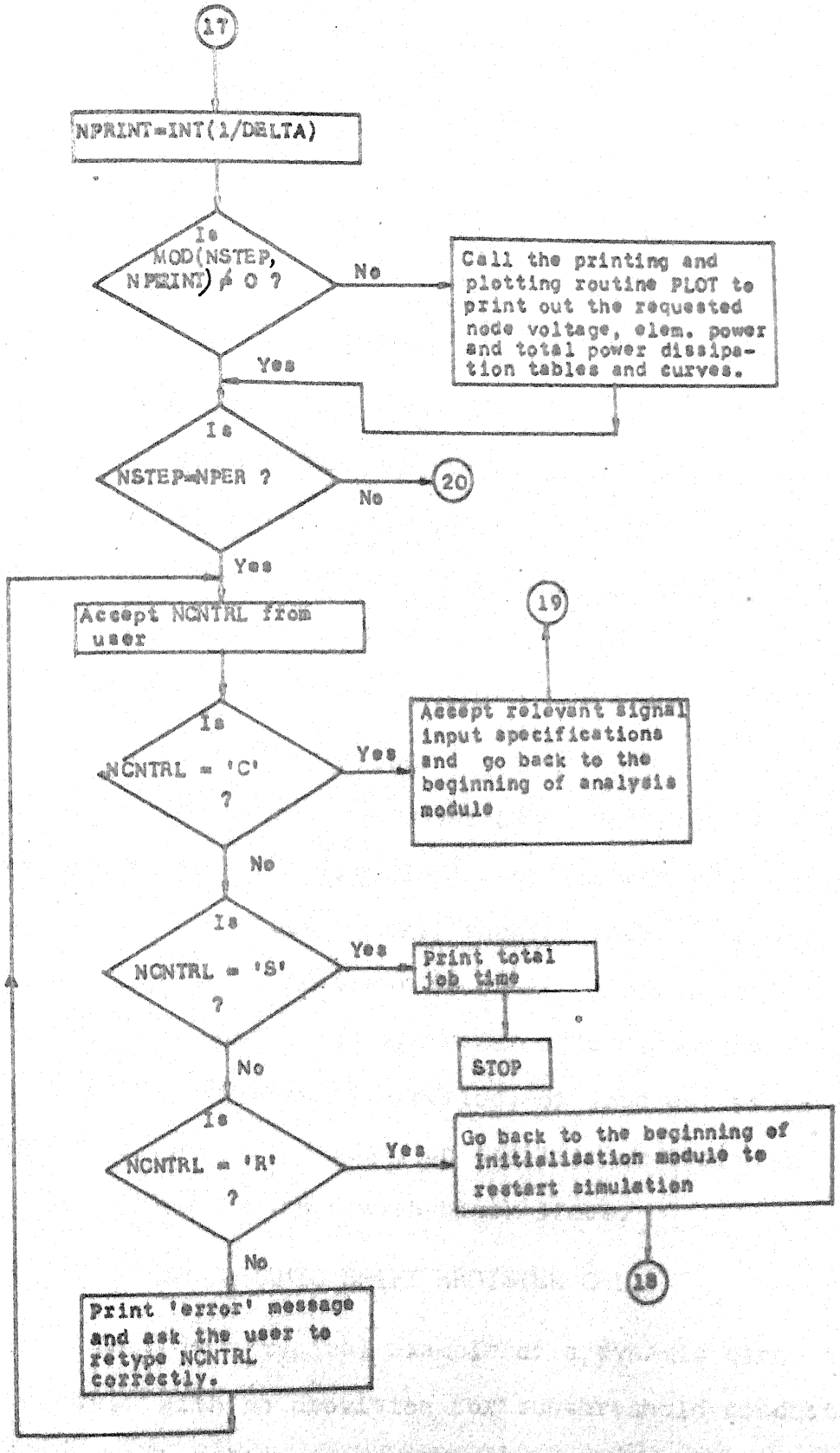


Fig. 3.17: Flow chart of output and control modules at the end of Analysis module.

CHAPTER 4

RESULTS

Different small and medium sized NMOS circuits have been analysed and the results compiled in this chapter. Comparison with results obtained by SPICE 2G shows relative improvement in simulation time, with increasing circuit size. The inaccuracies incurred by MOSIMR have also been made apparent by checking against the SPICE results.

In the voltage/power plots the points are joined with pencil for the monitored waveforms. The peaks (or humps) are marked with P's (red ink) and dips (or depressions) are marked with D's (green ink), showing their respective timings. Voltage levels of interest (such as final value reached or the tip of a hump etc.) are demarcated with blue lines. Finally any range of time/period, of interest (such as rise/fall time of an edge, an oscillation period or a pulse width etc.) are marked with black lines.

4.1 TWO PHASE DYNAMIC SHIFT REGISTER CELL:

This is a typical example of a dynamic circuit, driven by clock. With no provision for subthreshold conduction of 'off' pass transistors, MOSIMR can not figure out the true performance of such a dynamic circuit for slow clocking. So

we apply here fast clock and rapidly changing input to check the performance over 50 nsec., which is much less than the usual period of refreshing ($\sim$ msec.).

The voltage at node 3 follows the rising edge at 1 with a lag, to reach the final value $\sim$ 3.75V. This causes the inverter output to fall at node 4. The rise time of node 3 is 13 nsec. and fall time of node 4 is 10 nsec. When phase 2 of the clock at node 5 rises, pass transistor 2 turns 'on', tending to equalise the drain, source potentials by redistributing the charges on the two node capacitances. By initial state assignment the floating source node 6 is at logic low (i.e. 0.5V in MOSIMR and 0 V in SPICE) while the drain node 4 is at $\sim$.8V. Hence the driven end 4 is pulled down momentarily by the floating end (which is at a lower potential) when the clock input at node 5 rises, to produce a depression at D.

Comparing Figs. 4.1c and 4.1e the timings and voltage obtained by MOSIMR are found to be in agreement with that obtained by SPICE. The element power dissipation as calculated by MOSIMR over the simulation period have been printed and plotted for INVERTER 1 in table 4.1 and Fig. 4.1d. A sharp rise in power dissipation occurs between 4 nsec. and 10 nsec. as the inverter output makes a transition from high to low in this period.

For analysing this circuit (with 6 trans.) ratio of time taken by SPICE : time taken by MOSIMR = 4.79

4.2 NAND LATCH:

This is a typical example of a tightly coupled feed back circuit yielding a bistable element.

Initially node 3 is assigned 'low' and node 4 'high' (i.e. reset state). A low going pulse is applied at the input node 1 while node 2 is held constant 'high'. As the flip flop sets, outputs 3 and 4 are monitored and the waveforms roughly agree with that obtained by SPICE. The dip D occuring in V(3) at 31 nsec. and a corresponding peak P in V(4) (the peak is missing in SPICE plots because of voltage scale cramping) are caused by the rising transition of the pulse V(1). Fig. 4.2d shows the total power dissipated by the Latch plotted against time. The power peaks at P_1 and P_2 occur when both the NAND gates conduct under transient condition. The quescent power dissipation of 0.276 mw computed by MOSIMR, exactly coincides with that computed by SPICE (refer listing 4.2).

For analysing the Latch (a circuit with 6 trans.) the ratio of time taken by SPICE : time taken by MOSIMR = 4.08.

4.3 MOS RING OSCILLATOR:

This is a typical example of a circuit with a global feedback to yield an analog oscillator out of digital gates.

The oscillation is triggered by a falling edge at input 1 and the waveform at the feedback node, 4 is monitored.

The peaks and troughs of the periodic waveform at node 4 are 4.4V and 0.78V respectively in both MOSIMR and SPICE plots (Figs. 4.3c and 4.3e). However, the period of the waveform obtained by SPICE is $\sim$ 4.8 nsec. while that obtained by MOSIMR is $\sim$ 5.7 nsec. The discrepancy is due to the timing error, unavoidable in such nonunidirectional circuits (Sec. A.3.4). The timing error can accumulate over the period of simulation and result in considerable deviations as clearly shown in Figs. 4.3c and 4.3e. The MOSIMR plot completely misses the trough D_2 obtained by SPICE due to timing error of 1.5 nsec. cumulated over the simulation period of 15 nsec.

The waveform V(4) is near sinusoidal but with a slightly slower rise and sharper fall, caused by slow pull up through a load transistor of higher resistance and faster pull down through the low resistance driver transistor in inverter 2.

Power plot in Fig. 4.3d exhibits a period of oscillation $\sim$1.8 nsec. which is around $\frac{1}{3}$ of that of V(4).

For analysing the Ring oscillator (a circuit with 7 transistors) over 15 nsec., the ratio of time taken by SPICE : time taken by MOSIMR = 7.77.

4.4 ONE-BIT FULL ADDER WITH PASS-TRANSISTOR CARRY CHAIN:

This examplifies a typical combinatorial circuit whose response to the input waveforms, shown in Fig. 4.3a has been studied by monitoring the SUM (i.e. node 13 in MOSIMR circuit) and CARRY (i.e. node 9) outputs. The initial voltage assigned to V(9) which is at logic low is 0V in SPICE and 0.5V in MOSIMR. Discrepancy K observed in V(9) (Fig. 4.4d) is due to this difference in initial value of V(9). Further deviations can be noticed in the rise and fall slopes of the SUM and CARRY waveforms. However, the gross features, such as width of the SUM pulse (25 nsec. in MOSIMR and 27 nsec. in SPICE) and final voltages attained by SUM and CARRY in logic high/low states, appreciably agree.

The ringing observed at K in V(13) (Fig. 4.4c) is due to numerical over shoot (due to the sharp transition of the falling edge of the SUM pulse Sec. (2.4)), avoidable by lowering the time step of simulation.

The rise in CARRY waveform starts earlier than the fall in SUM waveform due to propagation delay in inverter 3 (Fig. 4.4a).

For analysing the Adder (a circuit with 21 transistors), the ratio of time taken by SPICE : time taken by MOSIMR = 9.69.

Listing of the Input files for this circuit accepted by MOSIMR has been given in Listing 4.4 to illustrate the user description of a circuit that does not contain any subcircuit.

4.5 TWO-BIT JOHNSON RING COUNTER:

This circuit has been chosen as a typical example of a finite state machine structure. It is a switched tail ring counter with 4 states. Analysis starts from the reset state (the counter is reset by making the reset input R = '1') recognised as Count = 0 and represented by the output states V(9) = '1' ; V(10) = '0'; V(11) = '0' ; V(12) = '0'. Thereafter the reset is removed by making R = '0' and one cycle of 2ϕ-clock is applied at nodes 1 and 2 (with ϕ_2 initially high) to effect transition to the next state i.e. Count = 1 (represented by V(9) = '0' ; V(10) = '1'; V(11) = '0' ; V(12) = '0') while V(9) and V(10) are monitored.

In the plots obtained from MOSIMR and SPICE (Fig. 4.5c and 4.5e respectively) V(10) is found to start changing earlier than V(9) as the count crosses over from '0' to '1'. This is due to the propagation delay introduced by NOR gate 2 in the subcircuit, BLOCK, between nodes 5 and 4. The rise/fall times of the monitored waveforms ($\sim$5 - 6 ns) and voltage levels ($\sim$0.58V when logic low and 5 V when logic high), obtained by MOSIMR and SPICE agree very satisfactorily.

A hump P in V(9) occurs between 49 ns - 53 ns, synchronised with the rising edge of ϕ_2. It is due to the momentary lowering of V(4) by the redistribution of charge between the subckt. nodes 10 and 4 as pass transistor 3 is turned 'on', in the subcircuit BLOCK. The voltage level and timing of the hump, given by MOSIMR roughly matches with that of SPICE.

Total power dissipation in the circuit is plotted against time by MOSIMR in Fig. 4.5d.

The input data for the circuit (which involves subckts.), as accepted by MOSIMR and SPICE are given side by side in listings 4.5a and 4.5b to make relative merits and demerits apparent. The format-less inputting of SPICE is more user oriented than the formatted input of MOSIMR. But absence of macro blocks, such as logic gates, in SPICE necessitates these to be described as subcircuits before being used in the main ckt. In some cases therefore, a SPICE input file may turn out to be more lengthy than that of MOSIMR. In the given example the input files are of roughly equal lengths.

For the analysis of this medium sized circuit with 52 transistors, the ratio of time required by SPICE : time required by MOSIMR = 13.6.

4.6 CIRCULATING REFRESH REGISTER WITH R/W CONTROL:

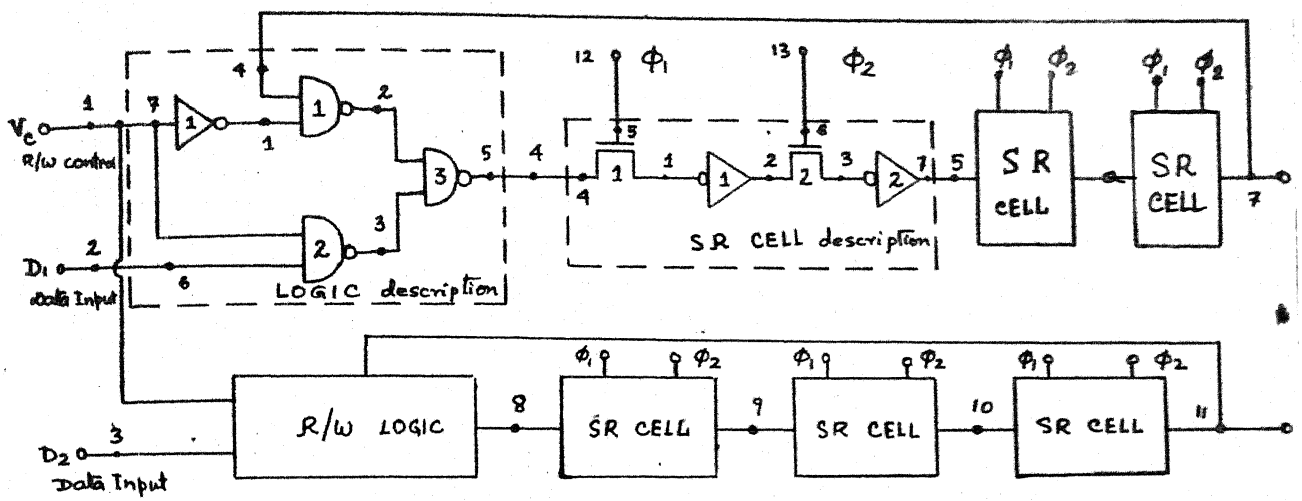
The refresh register is tested in the 'write' mode with control input V_c (i.e. input 1) held at '1'. Floating node 3 in subckt. CELL is assigned logic low initially. Data input to the upper bank of registers, D1 (i.e. input 2) is held constant at '0' and the data input D2 (i.e. input 3) is held constant at '1' so that only the bits in the SR cells belonging to the upper bank change with clocking, from 1 to 0, while the lower bank remains unchanged. One cycle of the 2 phase clock is applied at nodes 12 and 13 while nodes 5 and 9 are monitored. V(5) changes from logic high to logic low and V(9) remains constant, at logic high, during this clocking, as expected. The gross nature of the waveform V(5) (namely, the fall time, timing of the hump P or the steady value attained by V(5) at logic low ($\sim$ 0.7V)), given by MOSIMR tally with that obtained by SPICE.

However, the circuits analysed by SPICE and MOSIMR in this case though logically equivalent are not physically identical (refer Figs. 4.6a and 4.6b). In the circuit used by SPICE, the SR cell consists of pass transistors and inverters, whereas the SR cell, in the circuit used by MOSIMR, consists of pass transistors and inverting buffers. This results in deviations in the details of the voltage waveforms

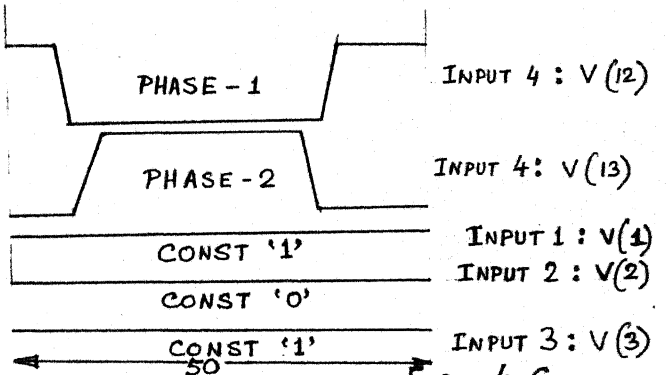
V(5) (e.g. SPICE, using inverters in place of inverting buffers in the SR-cells, gives a smaller hump at P compared to MOSIMR).

Though SPICE analyses a simpler circuit with 58 transistors and MOSIMR, a more complex circuit with 82 transistors, over a simulation period of 50 nsec., the ratio of time taken by SPICE : time taken by MOSIMR is 24.12.

This large gain in the speed of analysis by MOSIMR is due to detection of latency of the lower bank of registers. MOSIMR constrains its analysis to the upper bank of registers which undergo change and proceeds along the SR chain only upto that point which is active. Thus leaving out a major fraction of the circuit from computation. SPICE on the other hand does not exploit the inactivity of the whole lower bank of shift registers and performs the analysis blindly over the whole circuit in each time step.



CIRCULATING REGISTER – MOSIMR description



INPUT 4 : V(12)

INPUT 4: V(13)

INPUT 1 : V(1)

INPUT 2 : V(2)

INPUT 3 : V(3)

MONITOR : V(5), V(9)

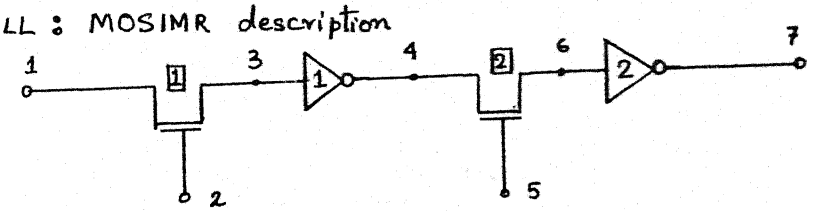
INITIAL STATE : V(3) = '0' in CELL.

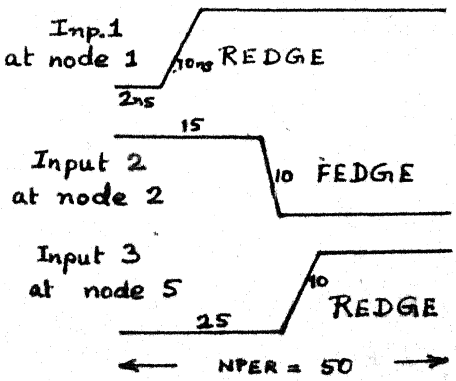
CIRCUIT SIZE : 82 transistors

FIG. 4.6a

INPUT WAVEFORMS

2φ DYNAMIC S-R CELL : MOSIMR description





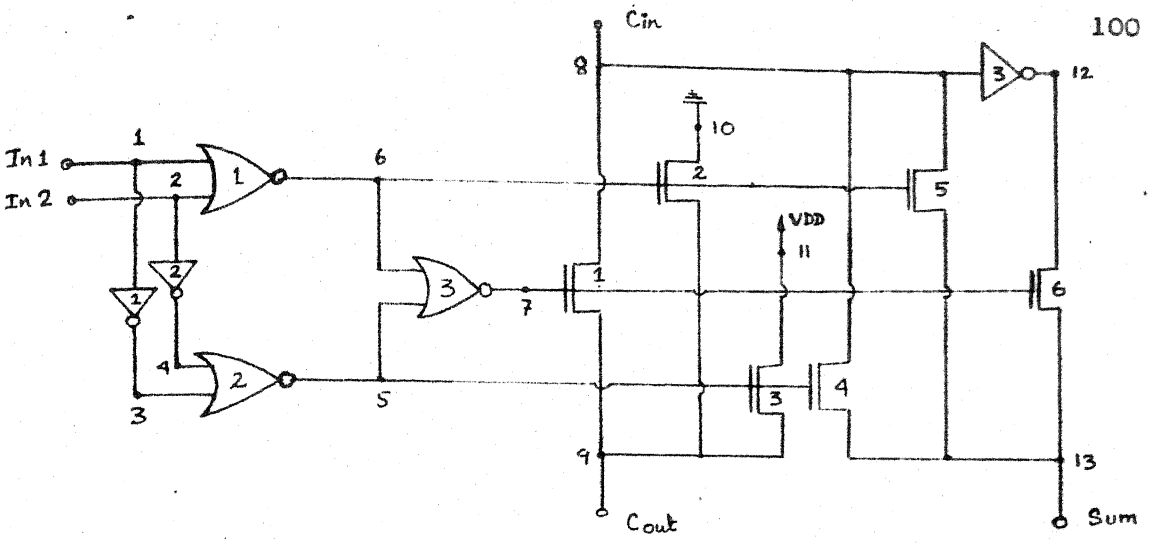
POWER OF INVERTER 1 & VOLTAGES AT NODES 3 & 4 MONITORED

INITIAL STATE : FLOATING NODE 6 AT '0'

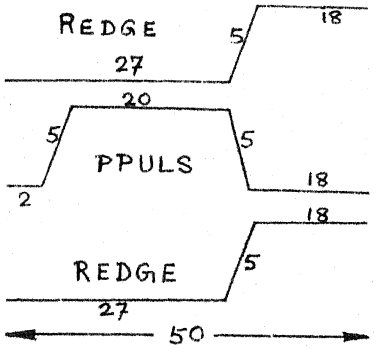
CIRCUIT SIZE : SIX TAANSISTORS

FIG. 4.1a

INPUT WAVEFORMS



One-bit FULL ADDER : MOSIMR description



INPUT 1 : V(1)

INPUT 2 : V(2)

INPUT 3 : V(8)

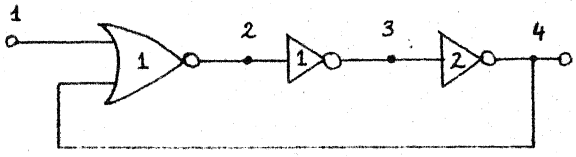
MONITOR : V(13), V(9)

INITIAL STATE : V(9) = 0.5V

CIRCUIT SIZE : 21 transistors

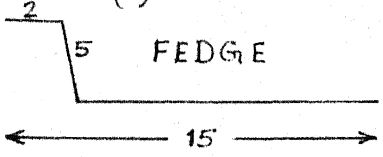
INPUT WAVEFORMS

FIG. 4.4a



MOS RING OSCILLATOR : MOSIMR description

INPUT 1 : V(1)

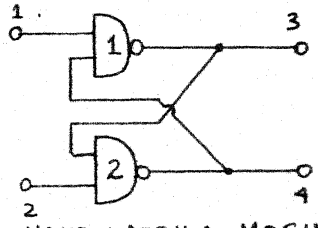


INPUT WAVEFORMS

MONITOR : V(4) & TOTAL POWER

CIRCUIT SIZE : 7 transistors

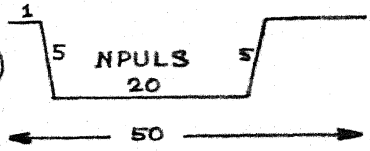
FIG. 4.3 a



NAND LATCH : MOSIMR description

INPUT 2 : V(2) CONST '1'

INPUT 1 : V(1)



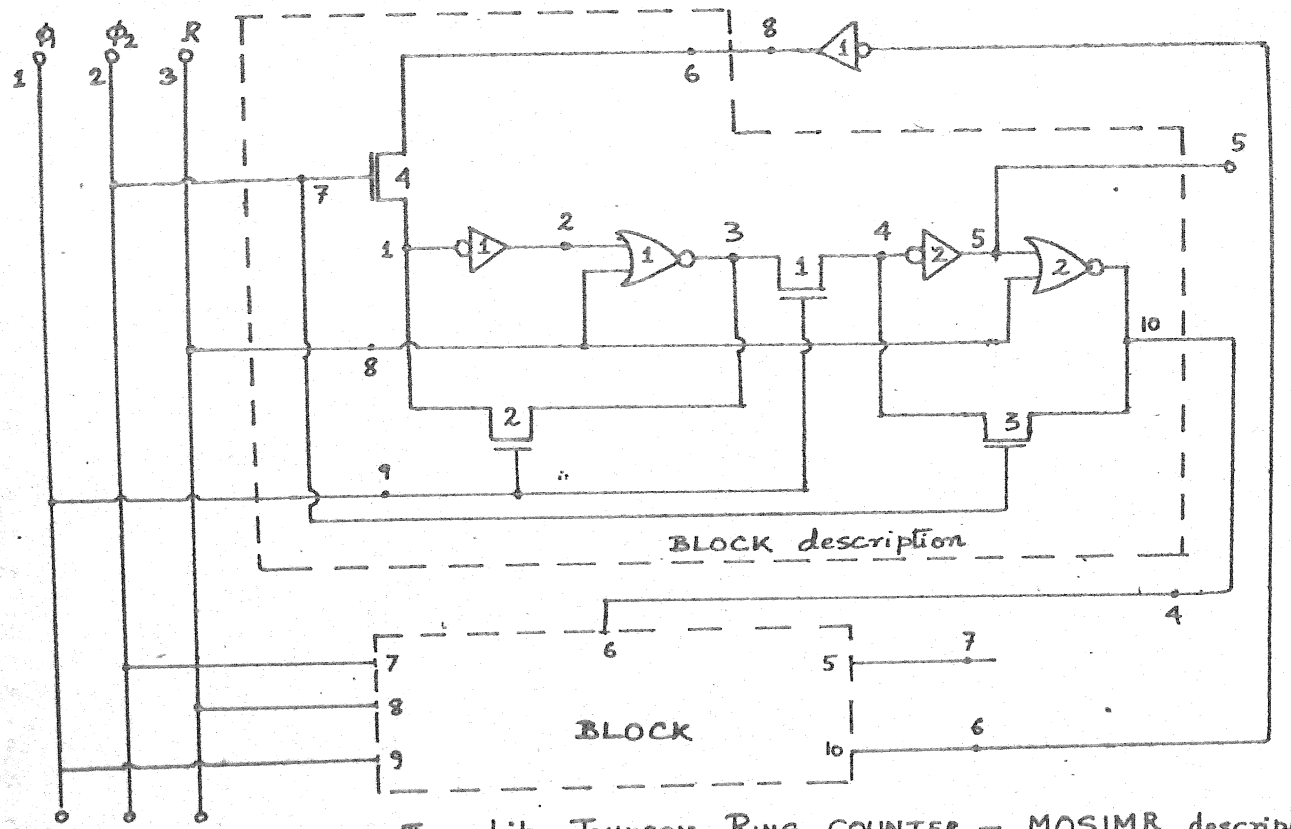
INPUT WAVEFORMS

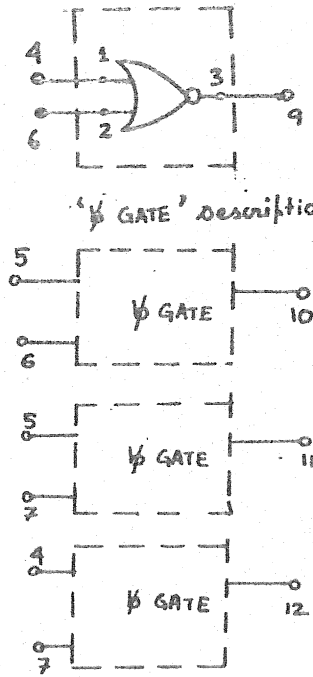
MONITOR : V(3), V(4) & TOTAL POWER

INITIAL STATE : V(3) = '0' ; V(4) = '1'

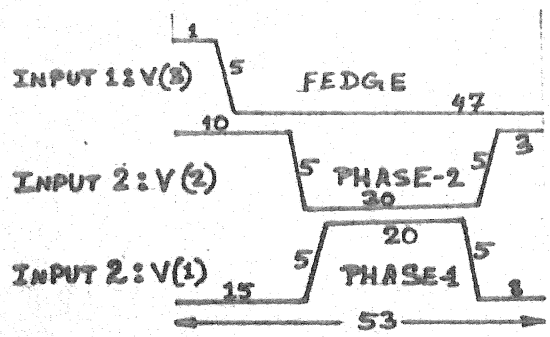
CIRCUIT SIZE : 6 transistors

FIG. 4.2a





Two-bit JOHNSON RING COUNTER – MOSIMR description



INPUT WAVEFORMS

MONITOR : V(9), V(10), Total power

INITIAL STATE : V(3) = '1' ; V(10) = '0' in CELL

CIRCUIT SIZE : 52 transistors

FIG. 4.5a

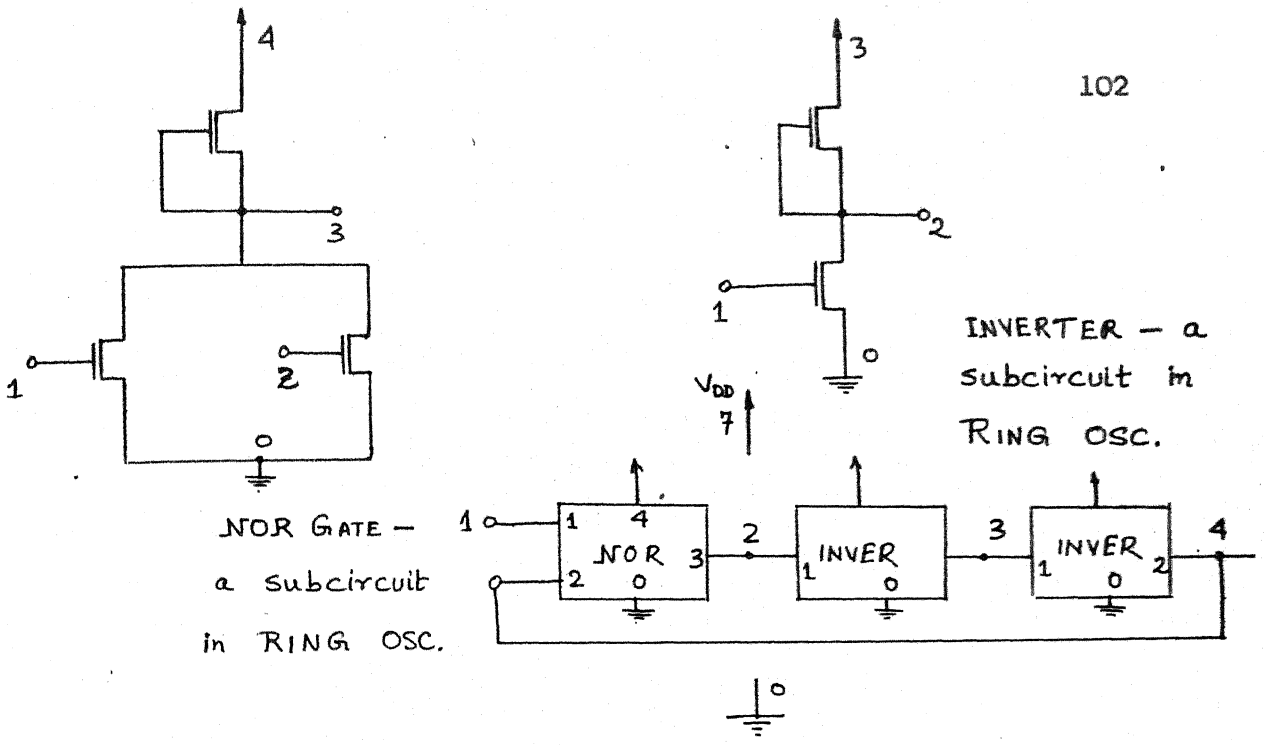
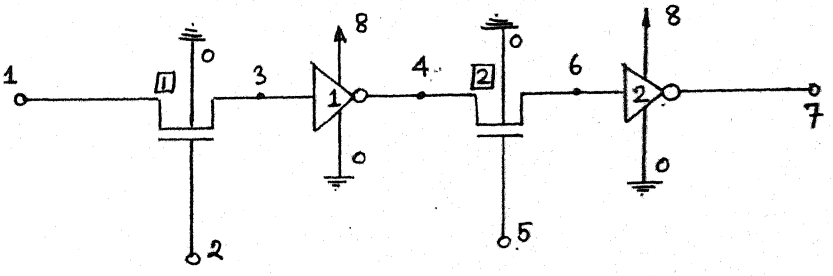
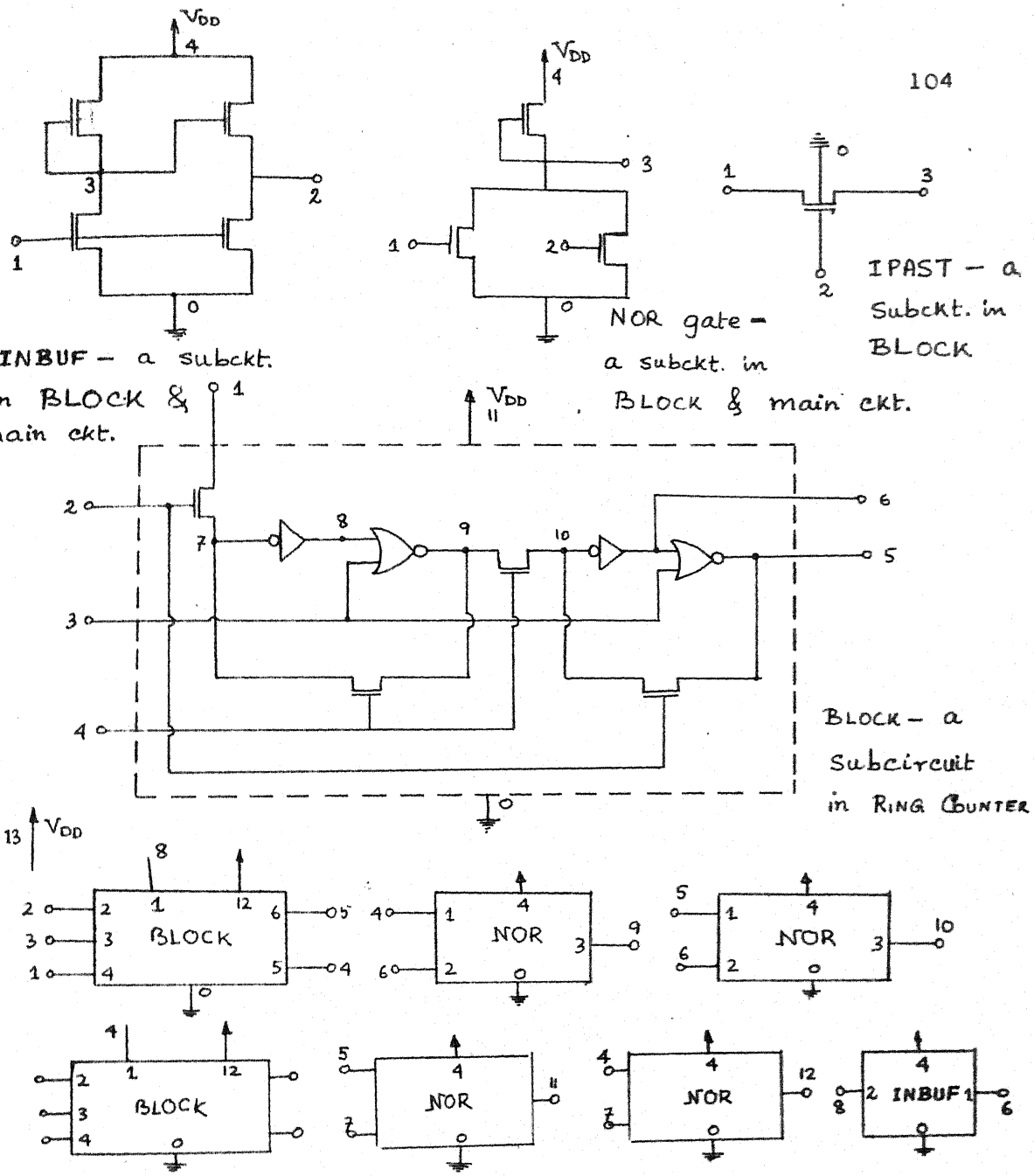


FIG. 4.3b



2φ DYNAMIC S-R CELL : SPICE description

FIG. 4.1b



RING COUNTER : SPICE description

Fig. 4.5 b

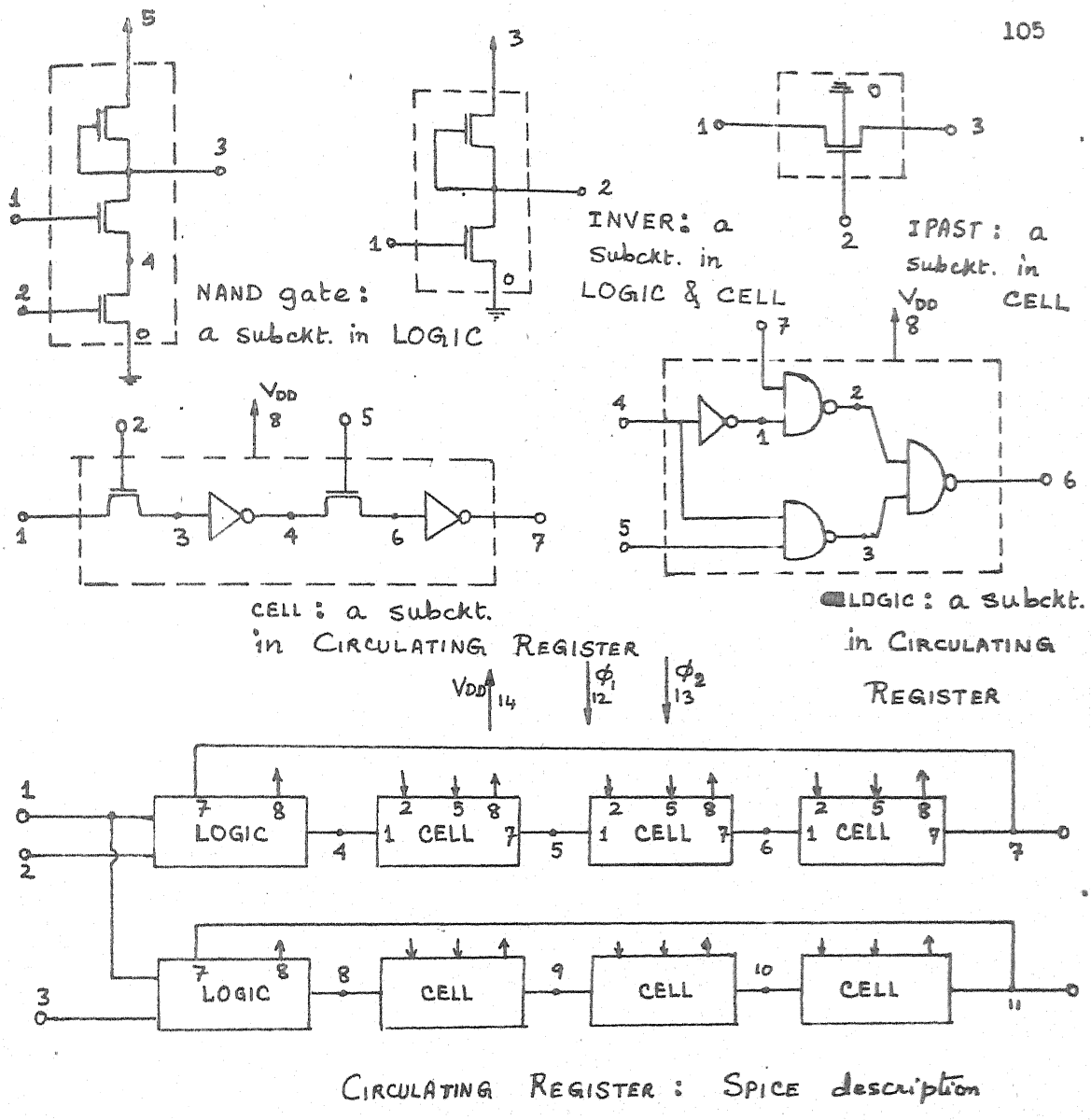

NAND gate: a subckt. in LOGIC
INVER: a Subckt. in LOGIC & CELL
IPAST : a subckt. in CELL
CELL : a subckt. in CIRCULATING REGISTER
LOGIC : a subckt. in CIRCULATING REGISTER
LOGIC
CELL
CIRCULATING REGISTER : SPICE description

FIG. 4.6b

```
* TWO PHASE DYNAMIC SHIFT REGISTER CELL *
*****************************************

VOLTAGE WAVEFORMS AT SPECIFIED NODES:                                  ------MOSIMR------

*****************************************

TIME(N-SEC)                                VOLTAGES(VOLTS)------>

          0.000E+00    0.100E+01    0.200E+01    0.300E+01    0.400E+01    0.500E+01
```

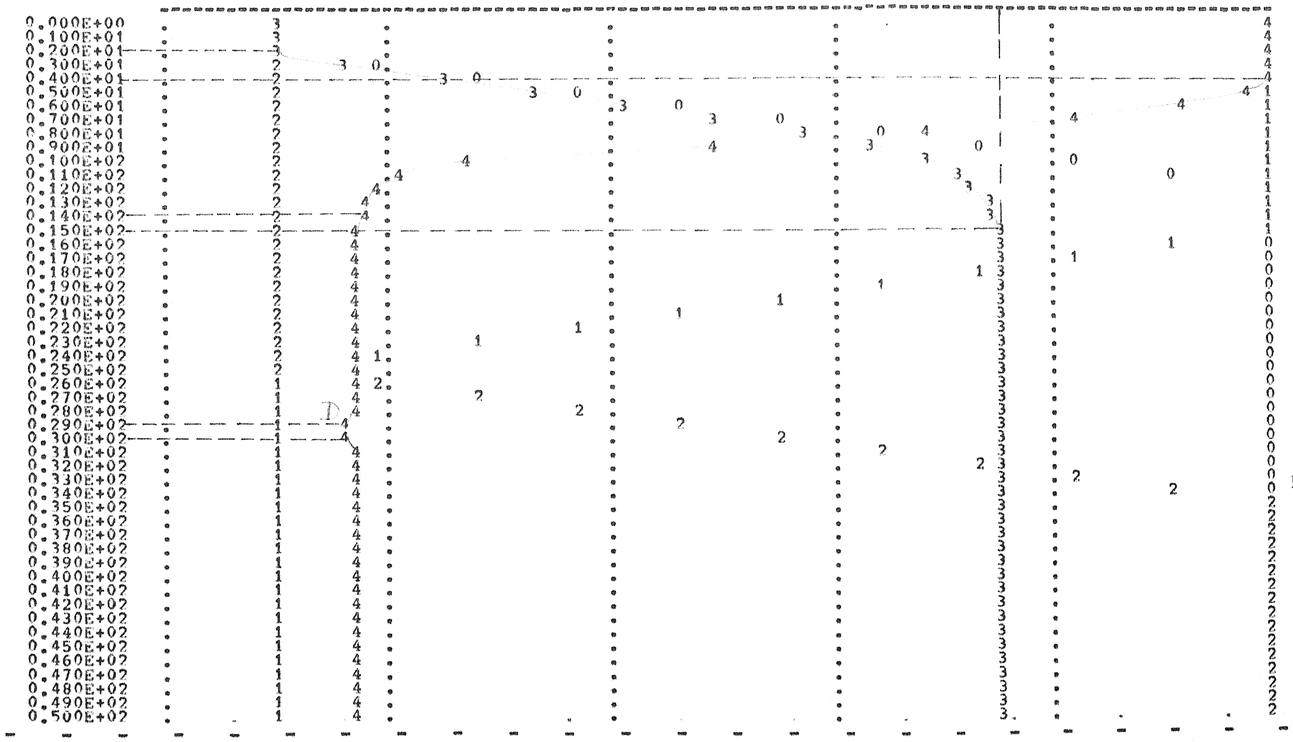


Figure 4.1(c)

```
JOB CONCLUDED                                          TOTAL JOB TIME=   1.08
```

* TWO PHASE DYNAMIC SHIFT REGISTER CELL *
****

TABLE II- ELEMENT POWERS VERSUS TIME :

****

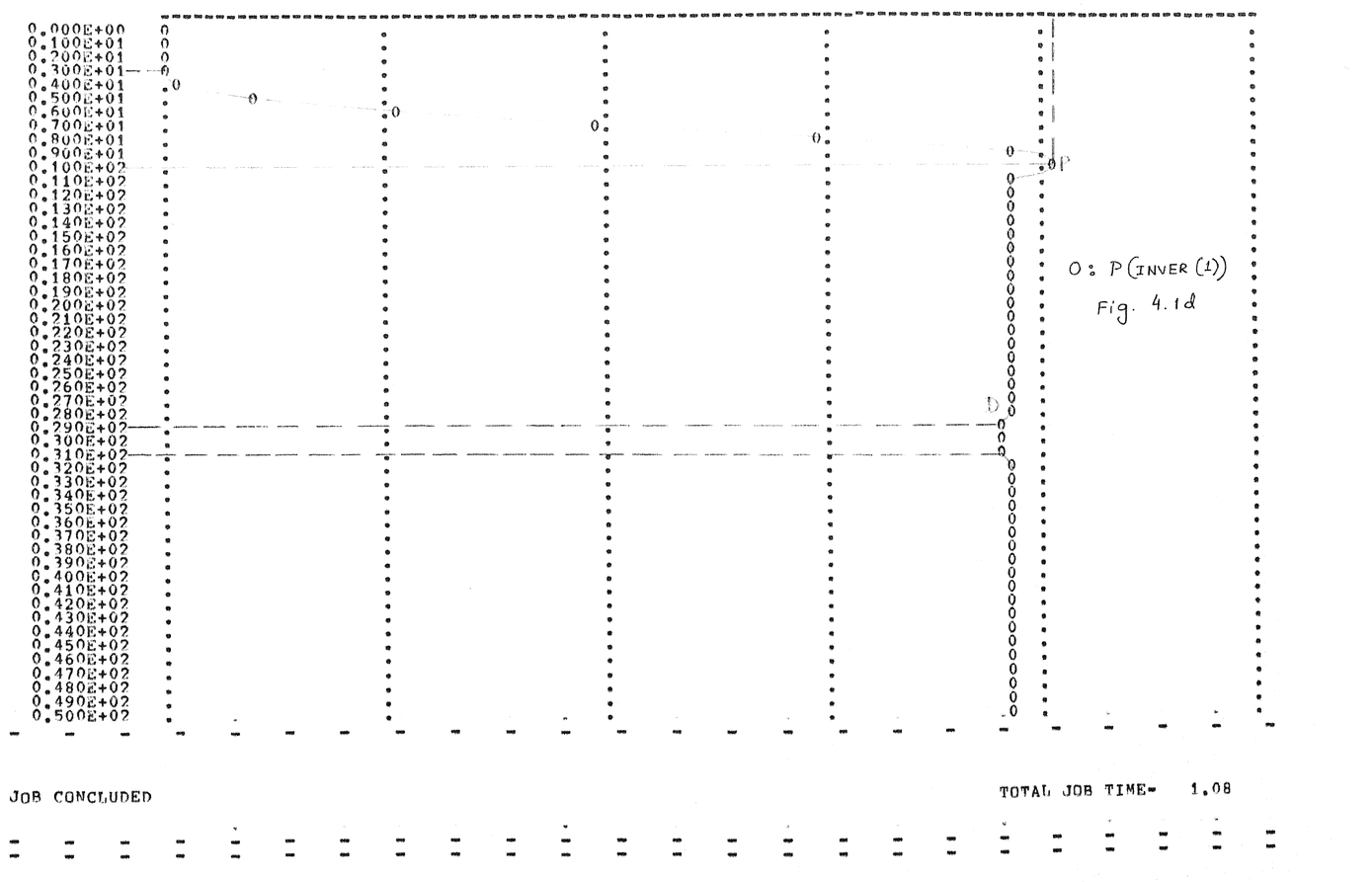
TIME(N-SEC)	ELEMENT POWERS(U-WATTS)
	INVER 1
0.000E+00	0.000E+00
0.100E+01	0.000E+00
0.200E+01	0.000E+00
0.300E+01	0.000E+00
0.400E+01	0.783E+01
0.500E+01	0.594E+02
0.600E+01	0.154E+03
0.700E+01	0.282E+03
0.800E+01	0.426E+03
0.900E+01	0.557E+03
0.100E+02	0.580E+03
0.110E+02	0.558E+03
0.120E+02	0.554E+03
0.130E+02	0.553E+03
0.140E+02	0.553E+03
0.150E+02	0.552E+03
0.160E+02	0.552E+03
0.170E+02	0.552E+03
0.180E+02	0.552E+03
0.190E+02	0.552E+03
0.200E+02	0.552E+03
0.210E+02	0.552E+03
0.220E+02	0.552E+03
0.230E+02	0.552E+03
0.240E+02	0.552E+03
0.250E+02	0.552E+03
0.260E+02	0.552E+03
0.270E+02	0.552E+03
0.280E+02	0.551E+03
0.290E+02	0.549E+03
0.300E+02	0.549E+03
0.310E+02	0.550E+03
0.320E+02	0.551E+03
0.330E+02	0.551E+03
0.340E+02	0.552E+03
0.350E+02	0.552E+03
0.360E+02	0.552E+03
0.370E+02	0.552E+03
0.380E+02	0.552E+03
0.390E+02	0.552E+03
0.400E+02	0.552E+03
0.410E+02	0.552E+03
0.420E+02	0.552E+03
0.430E+02	0.552E+03
0.440E+02	0.552E+03
0.450E+02	0.552E+03
0.460E+02	0.552E+03
0.470E+02	0.552E+03
0.480E+02	0.552E+03
0.490E+02	0.552E+03
0.500E+02	0.552E+03

Table : 4.1

JOB CONCLUDED

TOTAL JOB TIME= 1.08

```
* TWO PHASE DYNAMIC SHIFT REGISTER CELL *
*****************************************

                                                                    ------MOSIMR------

POWER WAVEFORMS FOR SPECIFIED ELEMENTS :

*****************************************

TIME(N-SEC)                          ELEMENT POWERS(U-WATTS) ----->

              0.000E+00    0.144E+03    0.288E+03    0.432E+03    0.576E+03    0.720E+03
```




```
JOB CONCLUDED                                        TOTAL JOB TIME-   1.08
```

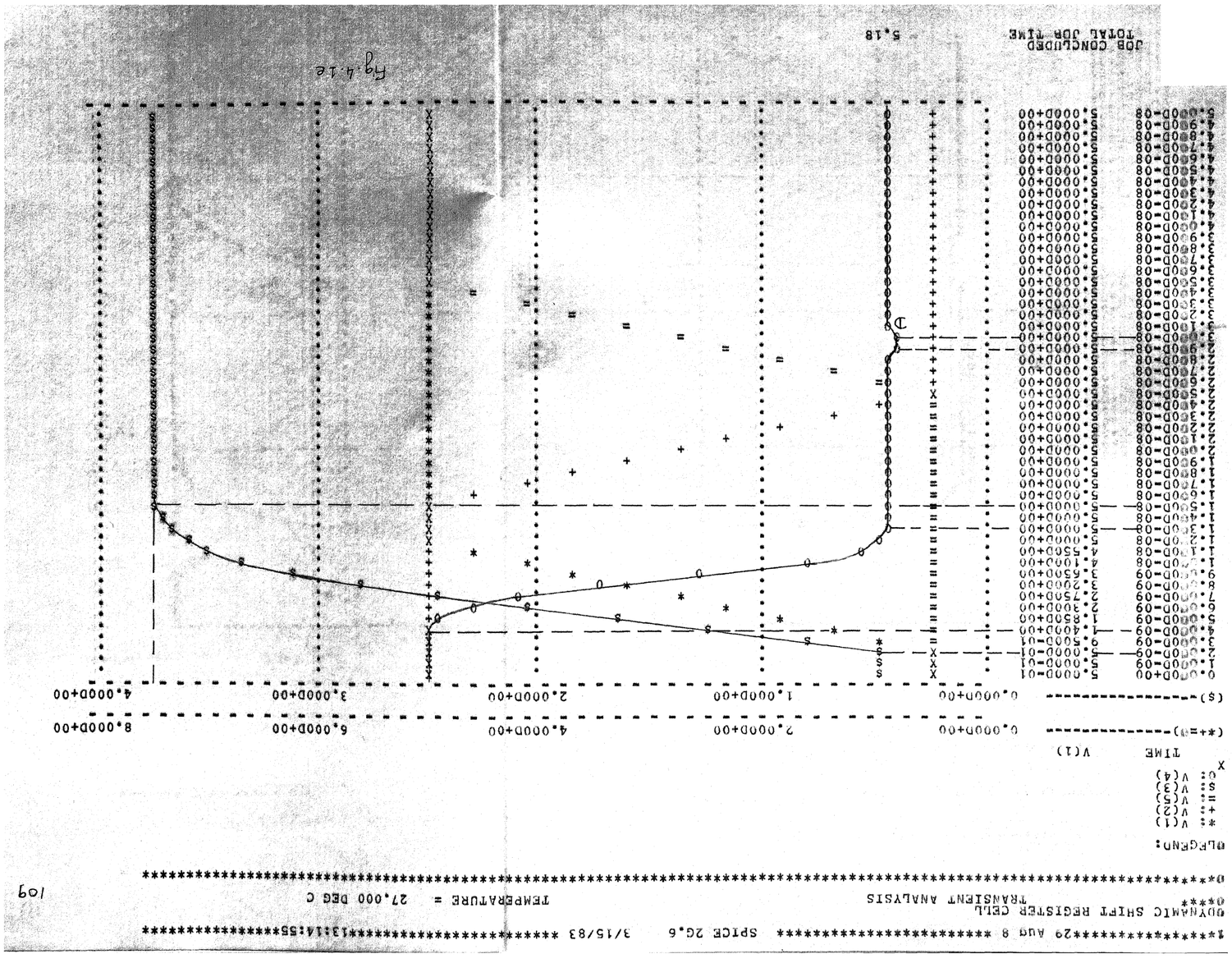

1*********29 Aug 8 ********** SPICE 2G.6 3/15/83 **********13:14:55**********
0*** DYNAMIC SHIFT REGISTER CELL TRANSIENT ANALYSIS TEMPERATURE = 27.000 DEG C
0**********
0LEGEND:
*: V(1)
+: V(2)
=: V(5)
S: V(3)
C: V(4)
X
TIME V(1)
(*+=)---------- 0.000D+00 2.000D+00 4.000D+00 6.000D+00 8.000D+00
($)---------- 0.000D+00 1.000D+00 2.000D+00 3.000D+00 4.000D+00
JOB CONCLUDED
TOTAL JOB TIME 5.18

Fig. 4.1e

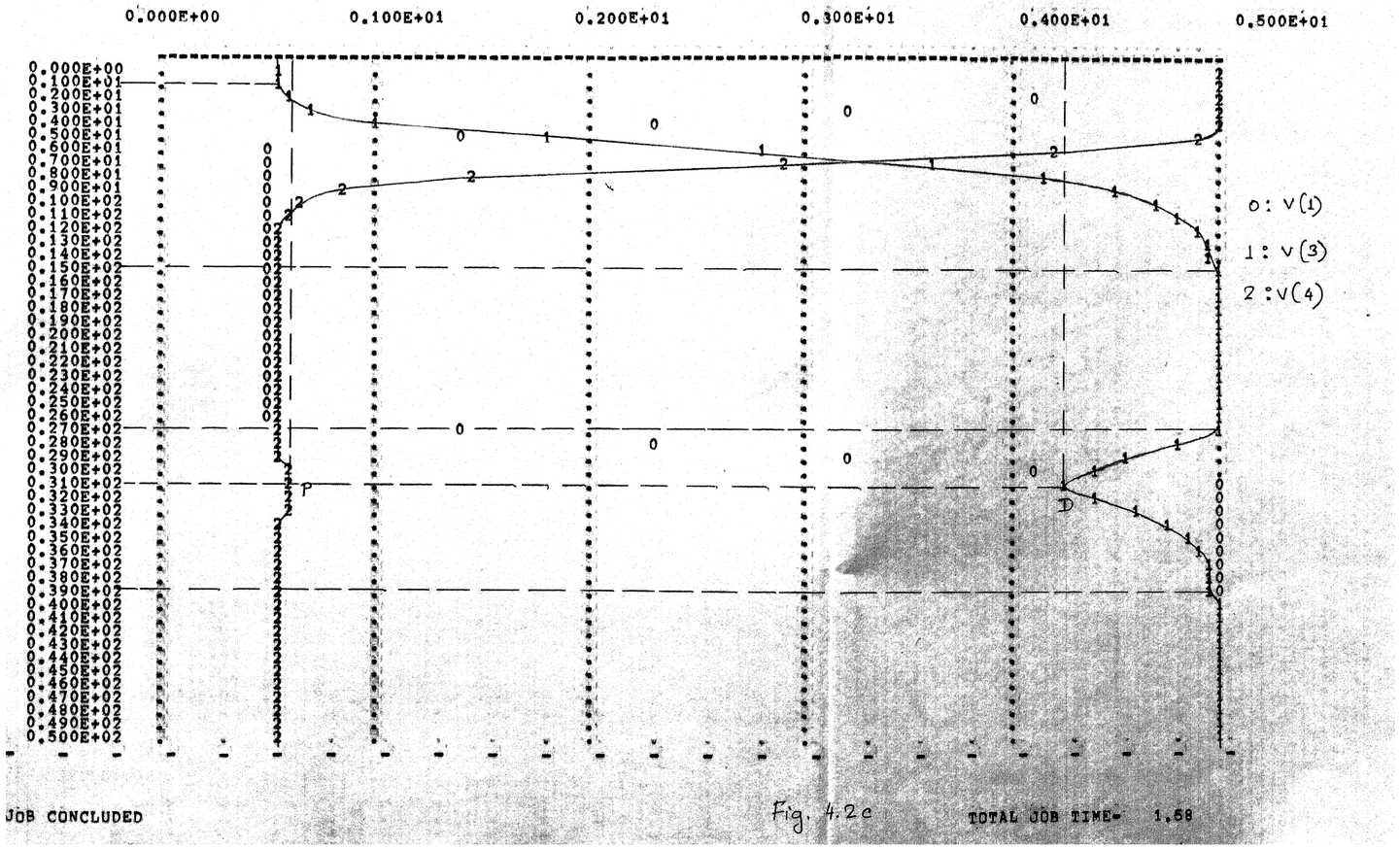
PERFORMANCE OF A NAND LATCH
****

VOLTAGE WAVEFORMS AT SPECIFIED NODES:

****

------MOSIMR------

TIME(N-SEC) VOLTAGES(VOLTS)----->



JOB CONCLUDED

Fig. 4.2c

TOTAL JOB TIME= 1.58

PERFORMANCE OF A NAND LATCH
****

TABLE II-TOTAL POWER VERSUS TIME
WAVEFORM OF TOTAL POWER DISSIPATED IN CKT. BLOCK :

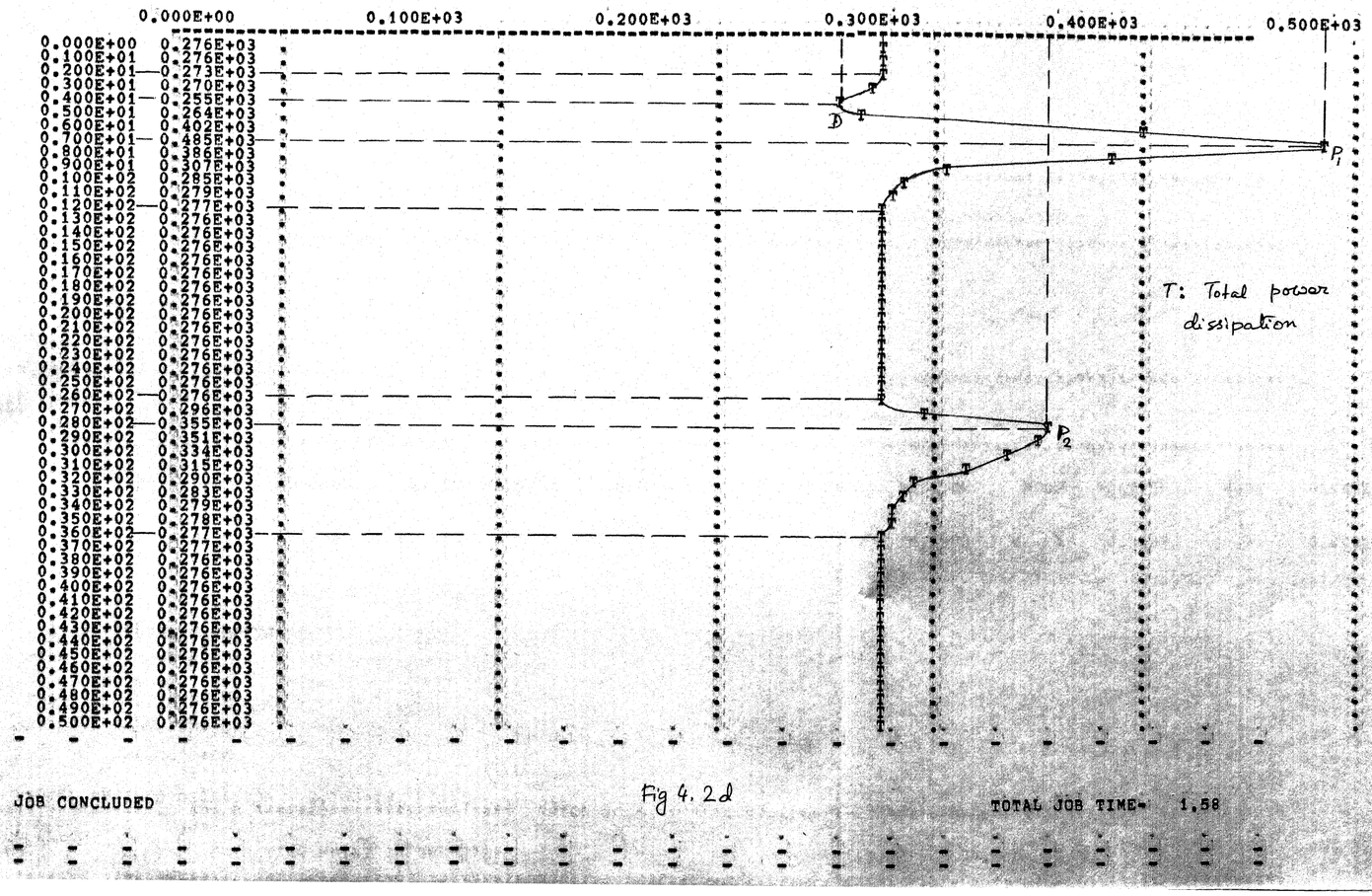
------MOSIMR------

****

TIME(N-SEC) TOTAL POWER

TOTAL POWER(U-WATTS)----->

TIME(N-SEC)	TOTAL POWER
0.000E+00	0.276E+03
0.100E+01	0.276E+03
0.200E+01	0.273E+03
0.300E+01	0.270E+03
0.400E+01	0.255E+03
0.500E+01	0.264E+03
0.600E+01	0.402E+03
0.700E+01	0.485E+03
0.800E+01	0.386E+03
0.900E+01	0.307E+03
0.100E+02	0.285E+03
0.110E+02	0.279E+03
0.120E+02	0.277E+03
0.130E+02	0.276E+03
0.140E+02	0.276E+03
0.150E+02	0.276E+03
0.160E+02	0.276E+03
0.170E+02	0.276E+03
0.180E+02	0.276E+03
0.190E+02	0.276E+03
0.200E+02	0.276E+03
0.210E+02	0.276E+03
0.220E+02	0.276E+03
0.230E+02	0.276E+03
0.240E+02	0.276E+03
0.250E+02	0.276E+03
0.260E+02	0.276E+03
0.270E+02	0.296E+03
0.280E+02	0.355E+03
0.290E+02	0.351E+03
0.300E+02	0.334E+03
0.310E+02	0.315E+03
0.320E+02	0.290E+03
0.330E+02	0.283E+03
0.340E+02	0.279E+03
0.350E+02	0.278E+03
0.360E+02	0.277E+03
0.370E+02	0.277E+03
0.380E+02	0.276E+03
0.390E+02	0.276E+03
0.400E+02	0.276E+03
0.410E+02	0.276E+03
0.420E+02	0.276E+03
0.430E+02	0.276E+03
0.440E+02	0.276E+03
0.450E+02	0.276E+03
0.460E+02	0.276E+03
0.470E+02	0.276E+03
0.480E+02	0.276E+03
0.490E+02	0.276E+03
0.500E+02	0.276E+03



JOB CONCLUDED

Fig 4.2d

TOTAL JOB TIME= 1.58

```
1*******23 Aug 8 ********  SPICE 2G.6    3/15/83 ********15:29:37*****
0NAND LATCH
0****     INPUT LISTING                   TEMPERATURE =   27.000 DEG C
0******************************************************************************


 * CKT DESCRIPTION
 .MODEL DRIVER NMOS VTO=1V KP=55.2E-6
 .MODEL LOAD NMOS VTO=-4V KP=6.9E-6
 .SUBCKT NAND 1 2 3 5
 ML 5 3 3 0 LOAD
 MD1 3 1 4 0 DRIVER
 MD2 4 2 0 0 DRIVER
 C1 1 0 .02PF
 C2 2 0 .02PF
 C3 3 0 .02PF
 C4 4 0 .02PF
 .ENDS
 X1 1 4 3 5 NAND
 X2 3 2 4 5 NAND
 * INPUTS
 VDD 5 0 DC 5
 V1 1 0 PULSE(5 .5 1NS 5NS 5NS 20NS 100NS)
 V2 2 0 DC 5
 * CONTROL
 .NODESET V(3)=.5 V(4)=5
 .TRAN 1NS 50NS
 * OUTPUT REQUESTS
 .PRINT TRAN V(1) V(3) V(4)
 .PLOT TRAN V(1) V(3) V(4)
 .END
1*****************23 Aug 8 ************************  SPICE 2G.6    3/15/83 ************************15:29:37*****************

0NAND LATCH
0****                  MOSFET MODEL PARAMETERS                          TEMPERATURE =   27.000 DEG C
0******************************************************************************************************************************


             DRIVER     LOAD
0TYPE        NMOS       NMOS
0LEVEL         1.000      1.000
0VTO           1.000     -4.000
0KP         5.52D-05   6.90D-06
1*****************23 Aug 8 ************************  SPICE 2G.6    3/15/83 ************************15:29:37*****************

0NAND LATCH
0****                  INITIAL TRANSIENT SOLUTION                       TEMPERATURE =   27.000 DEG C
0******************************************************************************************************************************

   NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE

(  1)    5.0000    (  2)    5.0000    (  3)    0.5357    (  4)    5.0000    (  5)    5.0000    (  6)    0.2583    (  7)    0.0000


     VOLTAGE SOURCE CURRENTS
     NAME         CURRENT

     VDD        -5.520D-05
     V1          0.000D+00
     V2          0.000D+00

     TOTAL POWER DISSIPATION   2.76D-04  WATTS
1*****************23 Aug 8 ************************  SPICE 2G.6    3/15/83 ************************15:29:37*****************

0NAND LATCH
0****                  OPERATING POINT INFORMATION                      TEMPERATURE =   27.000 DEG C
```

Listing: 4.2

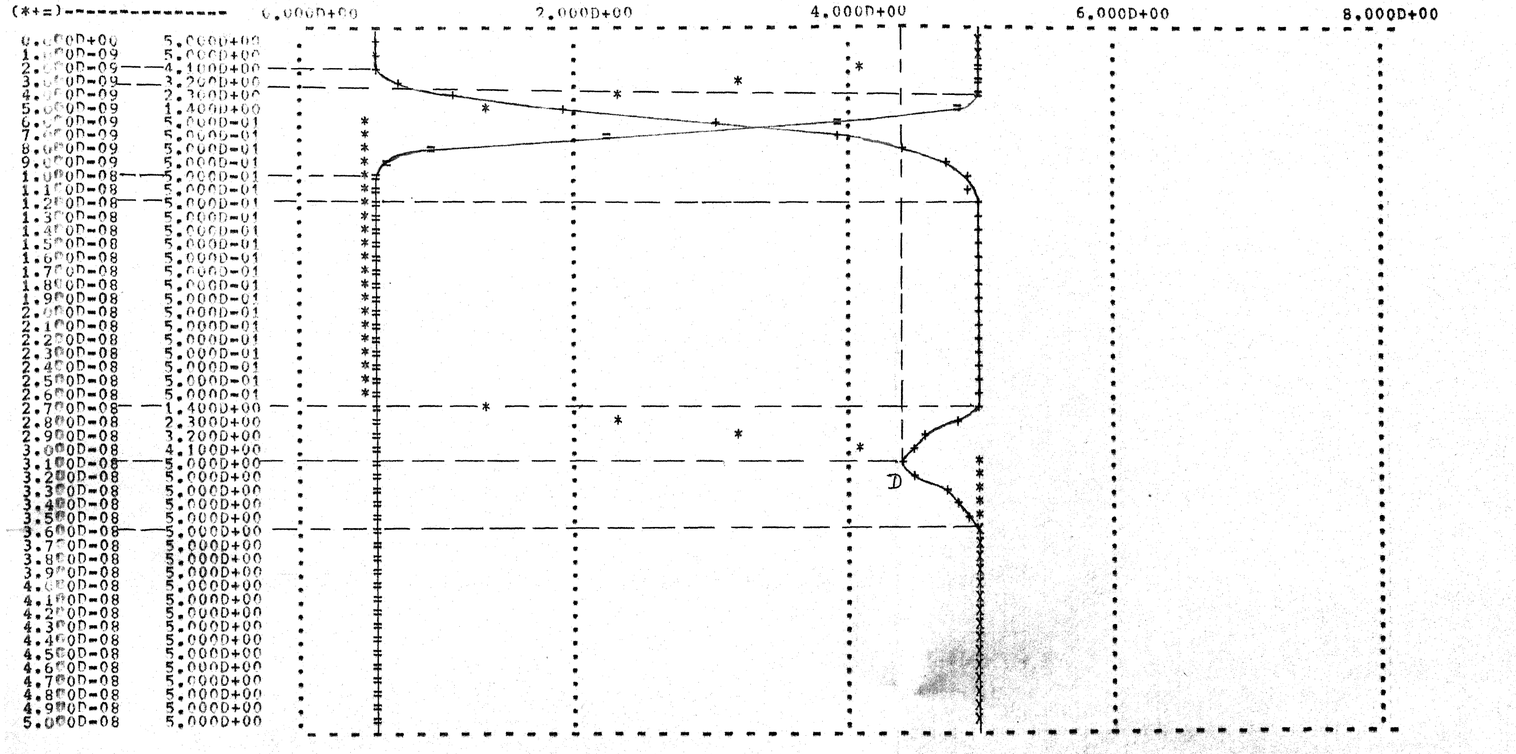
0LEGEND:

*: V(1)
+: V(3)
=: V(4)

X

TIME V(1)

(**=)---------- 0.000D+00 2.000D+00 4.000D+00 6.000D+00 8.000D+00



0.000D+00 5.000D+00
1.000D-09 5.000D+00
2.000D-09 4.100D+00
3.000D-09 3.200D+00
4.000D-09 2.300D+00
5.000D-09 1.400D+00
6.000D-09 5.000D-01
7.000D-09 5.000D-01
8.000D-09 5.000D-01
9.000D-09 5.000D-01
1.000D-08 5.000D-01
1.100D-08 5.000D-01
1.200D-08 5.000D-01
1.300D-08 5.000D-01
1.400D-08 5.000D-01
1.500D-08 5.000D-01
1.600D-08 5.000D-01
1.700D-08 5.000D-01
1.800D-08 5.000D-01
1.900D-08 5.000D-01
2.000D-08 5.000D-01
2.100D-08 5.000D-01
2.200D-08 5.000D-01
2.300D-08 5.000D-01
2.400D-08 5.000D-01
2.500D-08 5.000D-01
2.600D-08 5.000D-01
2.700D-08 1.400D+00
2.800D-08 2.300D+00
2.900D-08 3.200D+00
3.000D-08 4.100D+00
3.100D-08 5.000D+00
3.200D-08 5.000D+00
3.300D-08 5.000D+00
3.400D-08 5.000D+00
3.500D-08 5.000D+00
3.600D-08 5.000D+00
3.700D-08 5.000D+00
3.800D-08 5.000D+00
3.900D-08 5.000D+00
4.000D-08 5.000D+00
4.100D-08 5.000D+00
4.200D-08 5.000D+00
4.300D-08 5.000D+00
4.400D-08 5.000D+00
4.500D-08 5.000D+00
4.600D-08 5.000D+00
4.700D-08 5.000D+00
4.800D-08 5.000D+00
4.900D-08 5.000D+00
5.000D-08 5.000D+00

Y
0
0

JOB CONCLUDED
TOTAL JOB TIME 6.45

Fig. 4.2e

MOS RING OSCILLATOR
****

TABLE I - VOLTAGES VERSUS TIME :

------MOSIMR------

****

TIME(N-SEC) NODE VOLTAGES(VOLTS)

NODES=	1	2
0.000E+00	5.000	0.536
0.300E+00	5.000	0.536
0.600E+00	5.000	0.536
0.900E+00	5.000	0.536
0.120E+01	5.000	0.536
0.150E+01	5.000	0.536
0.180E+01	5.000	0.536
0.210E+01	4.910	0.536
0.240E+01	4.640	0.536
0.270E+01	4.370	0.536
0.300E+01	4.100	0.536
0.330E+01	3.830	0.536
0.360E+01	3.560	0.536
0.390E+01	3.290	0.536
0.420E+01	3.020	0.536
0.450E+01	2.750	0.536
0.480E+01	2.480	0.537
0.510E+01	2.210	0.541
0.540E+01	1.940	0.553
0.570E+01	1.670	0.577
0.600E+01	1.400	0.626
0.630E+01	1.130	0.714
0.660E+01	0.860	0.876
0.690E+01	0.590	1.202
0.720E+01	0.500	1.764
0.750E+01	0.500	2.379
0.780E+01	0.500	2.935
0.810E+01	0.500	3.405
0.840E+01	0.500	3.788
0.870E+01	0.500	4.088
0.900E+01	0.500	4.276
0.930E+01	0.500	4.278
0.960E+01	0.500	4.042
0.990E+01	0.500	3.566
0.102E+02	0.500	2.872
0.105E+02	0.500	2.048
0.108E+02	0.500	1.401
0.111E+02	0.500	1.026
0.114E+02	0.500	0.861
0.117E+02	0.500	0.835
0.120E+02	0.500	0.923
0.123E+02	0.500	1.162
0.126E+02	0.500	1.653
0.129E+02	0.500	2.258
0.132E+02	0.500	2.829
0.135E+02	0.500	3.317
0.138E+02	0.500	3.717
0.141E+02	0.500	4.033
0.144E+02	0.500	4.245
0.147E+02	0.500	4.281
0.150E+02	0.500	4.082

Table: 4.3a

JOB CONCLUDED

TOTAL JOB TIME= 0.78

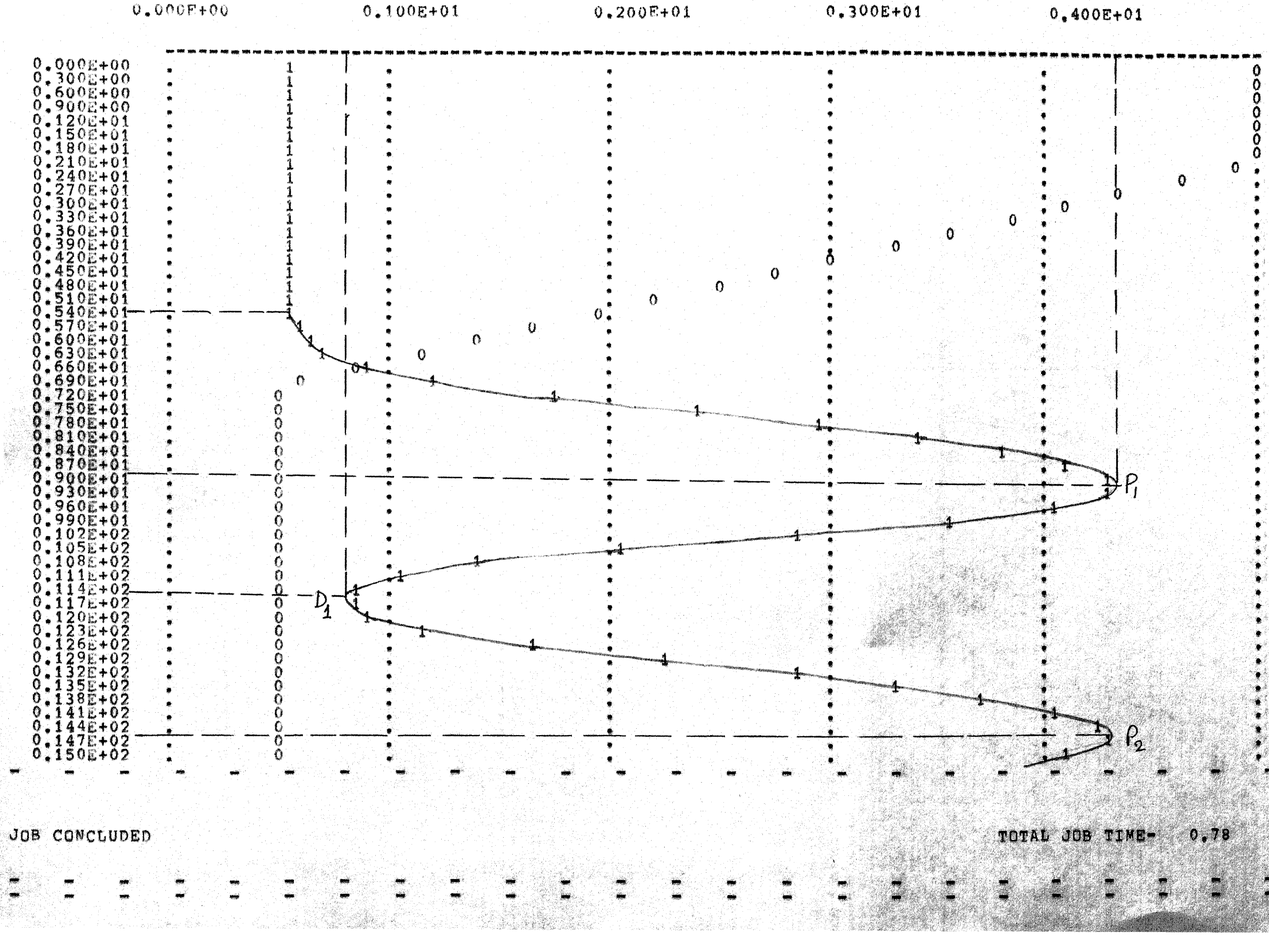
MOS RING OSCILLATOR
****

VOLTAGE WAVEFORMS AT SPECIFIED NODES:

------MOSIMR------

****

TIME(N-SEC) VOLTAGES(VOLTS)----->



0: V(1)
1: V(4)

Fig. 4.3c

JOB CONCLUDED

TOTAL JOB TIME= 0.78

MOS RING OSCILLATOR
****

TABLE II-TOTAL POWER VERSUS TIME
WAVEFORM OF TOTAL POWER DISSIPATED IN CKT. BLOCK :

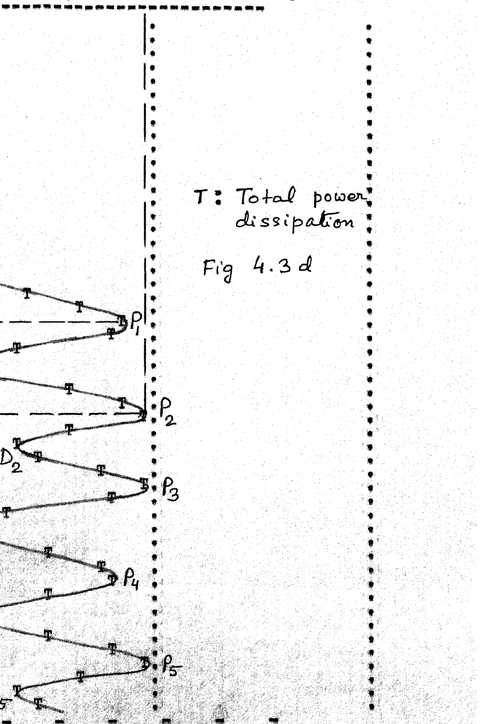
------MOSIMR------

****

TIME(N-SEC)	TOTAL POWER
0.000E+00	0.110E+04
0.300E+00	0.110E+04
0.600E+00	0.110E+04
0.900E+00	0.110E+04
0.120E+01	0.110E+04
0.150E+01	0.110E+04
0.180E+01	0.110E+04
0.210E+01	0.110E+04
0.240E+01	0.110E+04
0.270E+01	0.110E+04
0.300E+01	0.110E+04
0.330E+01	0.109E+04
0.360E+01	0.109E+04
0.390E+01	0.108E+04
0.420E+01	0.108E+04
0.450E+01	0.107E+04
0.480E+01	0.106E+04
0.510E+01	0.105E+04
0.540E+01	0.108E+04
0.570E+01	0.114E+04
0.600E+01	0.122E+04
0.630E+01	0.132E+04
0.660E+01	0.139E+04
0.690E+01	0.137E+04
0.720E+01	0.120E+04
0.750E+01	0.111E+04
0.780E+01	0.117E+04
0.810E+01	0.129E+04
0.840E+01	0.139E+04
0.870E+01	0.142E+04
0.900E+01	0.129E+04
0.930E+01	0.121E+04
0.960E+01	0.125E+04
0.990E+01	0.135E+04
0.102E+02	0.142E+04
0.105E+02	0.136E+04
0.108E+02	0.119E+04
0.111E+02	0.112E+04
0.114E+02	0.117E+04
0.117E+02	0.126E+04
0.120E+02	0.135E+04
0.123E+02	0.137E+04
0.126E+02	0.125E+04
0.129E+02	0.114E+04
0.132E+02	0.116E+04
0.135E+02	0.126E+04
0.138E+02	0.137E+04
0.141E+02	0.142E+04
0.144E+02	0.131E+04
0.147E+02	0.121E+04
0.150E+02	0.124E+04

TOTAL POWER(U-WATTS)----->

0.000E+00 0.360E+03 0.720E+03 0.108E+04 0.144E+04 0.180E+04



T: Total power dissipation

Fig 4.3 d

JOB CONCLUDED

TOTAL JOB TIME= 0.78

```
1***************28 Aug 8 ********************** SPICE 2G.6    3/15/83 ***********************12:13:33****************
0CMOS RING OSCILLATOR
0****                    TRANSIENT ANALYSIS                                    TEMPERATURE =   27.000 DEG C
0*************************************************************************************************************
```

X	TIME	V(1)	V(2)
	0.000E+00	5.000E+00	5.359E-01
	3.000E-10	5.000E+00	5.359E-01
	6.000E-10	5.000E+00	5.359E-01
	9.000E-10	5.000E+00	5.359E-01
	1.200E-09	5.000E+00	5.359E-01
	1.500E-09	5.000E+00	5.359E-01
	1.800E-09	5.000E+00	5.359E-01
	2.100E-09	4.910E+00	5.359E-01
	2.400E-09	4.640E+00	5.359E-01
	2.700E-09	4.370E+00	5.359E-01
	3.000E-09	4.100E+00	5.359E-01
	3.300E-09	3.830E+00	5.359E-01
	3.600E-09	3.560E+00	5.359E-01
	3.900E-09	3.290E+00	5.359E-01
	4.200E-09	3.020E+00	5.359E-01
	4.500E-09	2.750E+00	5.362E-01
	4.800E-09	2.480E+00	5.380E-01
	5.100E-09	2.210E+00	5.441E-01
	5.400E-09	1.940E+00	5.605E-01
	5.700E-09	1.670E+00	5.971E-01
	6.000E-09	1.400E+00	6.716E-01
	6.300E-09	1.130E+00	8.219E-01
	6.600E-09	8.600E-01	1.150E+00
	6.900E-09	5.900E-01	1.731E+00
	7.200E-09	5.000E-01	2.451E+00
	7.500E-09	5.000E-01	3.098E+00
	7.800E-09	5.000E-01	3.630E+00
	8.100E-09	5.000E-01	4.032E+00
	8.400E-09	5.000E-01	4.274E+00
	8.700E-09	5.000E-01	4.268E+00
	9.000E-09	5.000E-01	3.932E+00
	9.300E-09	5.000E-01	3.249E+00
	9.600E-09	5.000E-01	2.335E+00
	9.900E-09	5.000E-01	1.493E+00
	1.020E-08	5.000E-01	9.833E-01
	1.050E-08	5.000E-01	7.960E-01
	1.080E-08	5.000E-01	8.140E-01
	1.110E-08	5.000E-01	1.007E+00
	1.140E-08	5.000E-01	1.445E+00
	1.170E-08	5.000E-01	2.098E+00
	1.200E-08	5.000E-01	2.787E+00
	1.230E-08	5.000E-01	3.380E+00
	1.260E-08	5.000E-01	3.848E+00
	1.290E-08	5.000E-01	4.177E+00
	1.320E-08	5.000E-01	4.304E+00
	1.350E-08	5.000E-01	4.133E+00
	1.380E-08	5.000E-01	3.615E+00
	1.410E-08	5.000E-01	2.781E+00
	1.440E-08	5.000E-01	1.858E+00
	1.470E-08	5.000E-01	1.176E+00
	1.500E-08	5.000E-01	8.253E-01

Table 4.3b

```
1**********28 Aug 8 ********************** SPICE 2G.6    3/15/83 ********************12:13:33***************
0MOS RING OSCILLATOR
0****     TRANSIENT ANALYSIS                                    TEMPERATURE =   27.000 DEG C
0*************************************************************************************************************

0LEGEND:
 *: V(1)
 +: V(2)
X
     TIME        V(1)
(*+)----------  0.000D+00          2.000D+00          4.000D+00          6.000D+00          8.000D+00
0.000D+00    5.000D+00
3.000D-10    5.000D+00
6.000D-10    5.000D+00
9.000D-10    5.000D+00
1.200D-09    5.000D+00
1.500D-09    5.000D+00
1.800D-09    5.000D+00
2.100D-09    4.910D+00
2.400D-09    4.640D+00
2.700D-09    4.370D+00
3.000D-09    4.100D+00
3.300D-09    3.830D+00
3.600D-09    3.560D+00
3.900D-09    3.290D+00
4.200D-09    3.020D+00
4.500D-09    2.750D+00
4.800D-09    2.480D+00
5.100D-09    2.210D+00
5.400D-09    1.940D+00
5.700D-09    1.670D+00
6.000D-09    1.400D+00
6.300D-09    1.130D+00
6.600D-09    8.600D-01
6.900D-09    5.900D-01
7.200D-09    5.000D-01
7.500D-09    5.000D-01
7.800D-09    5.000D-01
8.100D-09    5.000D-01
8.400D-09    5.000D-01
8.700D-09    5.000D-01
9.000D-09    5.000D-01
9.300D-09    5.000D-01
9.600D-09    5.000D-01
9.900D-09    5.000D-01
1.020D-08    5.000D-01
1.050D-08    5.000D-01
1.080D-08    5.000D-01
1.110D-08    5.000D-01
1.140D-08    5.000D-01
1.170D-08    5.000D-01
1.200D-08    5.000D-01
1.230D-08    5.000D-01
1.260D-08    5.000D-01
1.290D-08    5.000D-01
1.320D-08    5.000D-01
1.350D-08    5.000D-01
1.380D-08    5.000D-01
1.410D-08    5.000D-01
1.440D-08    5.000D-01
1.470D-08    5.000D-01
1.500D-08    5.000D-01
Y
0
0
   JOB CONCLUDED
   TOTAL JOB TIME            6.06
```

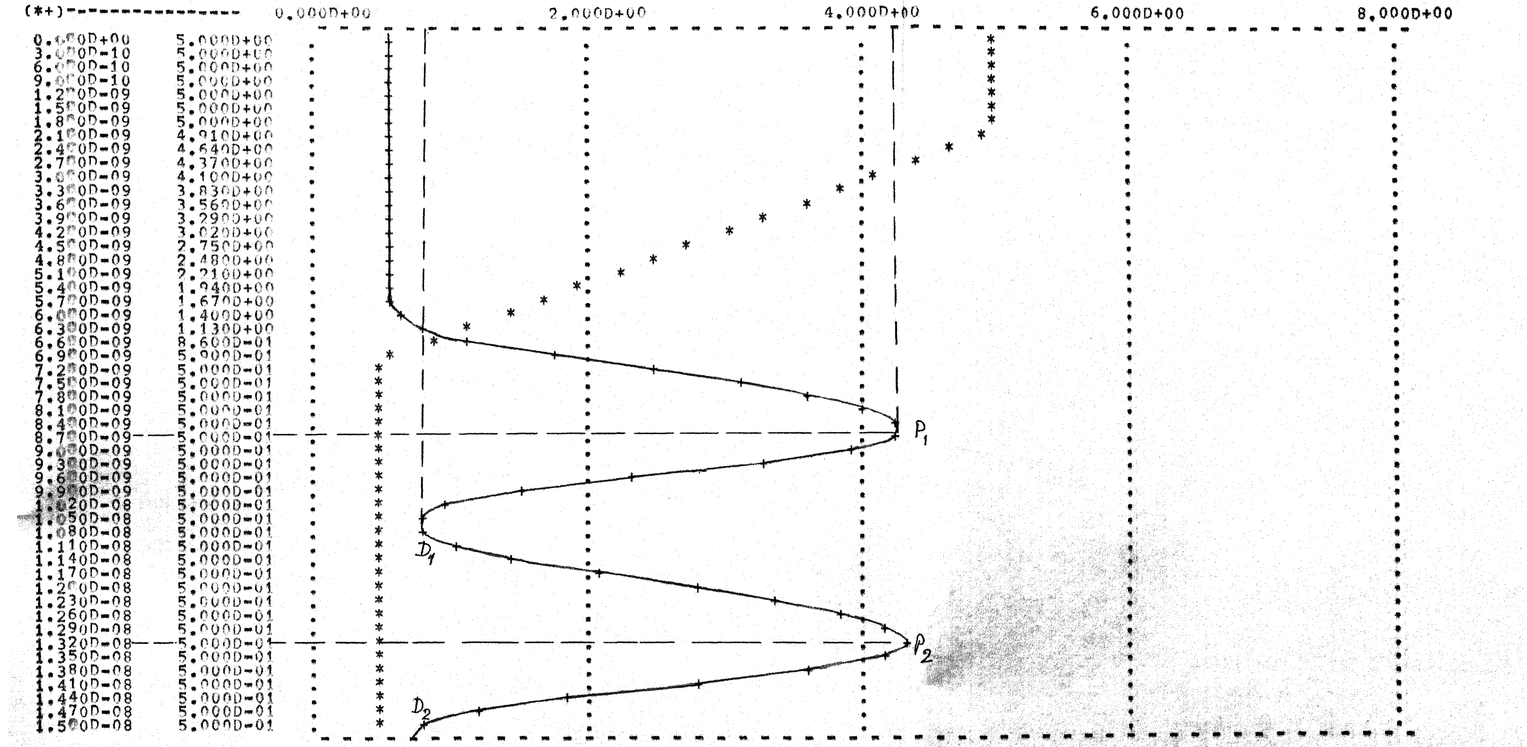


Fig. 4.3e

```
 CIRCUIT DESCRIPTION FILE - CKT.DES
ONE BIT FULL ADDER WITH PASS TRANSISTOR CARRY CHAIN
013 000
INVER 001
001 003 NOELM 000 NORG2 002
INVER 002
002 004 NOELM 000 NORG2 002
NORG2 001
001 002 006 INVER 001 INVER 002 NORG2 003
NORG2 002
004 003 005 NOELM 000 NOELM 000 NORG2 003
NORG2 003
006 005 007 IPAST 002 IPAST 003 IPAST 001
IPAST 001
008 007 009 INVER 003 IPAST 006 NOELM 000
IPAST 002
010 006 009 NOELM 000 IPAST 005 NOELM 000
IPAST 003
011 005 009 NOELM 000 IPAST 004 NOELM 000
IPAST 004
008 005 013 NOELM 000 NOELM 000 NOELM 000
IPAST 005
008 006 013 IPAST 004 NOELM 000 NOELM 000
IPAST 006
012 007 013 NOELM 000 NOELM 000 NOELM 000
INVER 003
008 012 IPAST 005 IPAST 006
INPUT 001
001 INGEN NORG2 001
INPUT 002
002 INGEN NORG2 001
INPUT 003
008 INGEN IPAST 001
INPUT 004
010 INGEN IPAST 002
INPUT 005
011 INGEN IPAST 003
CLOSE

C  CIRCUIT INITIALISATION FILE - INIT.CON
000
005
001 0
002 0
008 0
010 0
011 1
005
001 002 008 013 009
N 000
000
000
0. 0.
```

Listing : 4.4

ONE BIT FULL ADDER WITH PASS TRANSISTOR CARRY CHAIN
****

VOLTAGE WAVEFORMS AT SPECIFIED NODES:

****

TIME(N-SEC) VOLTAGES(VOLTS)----->

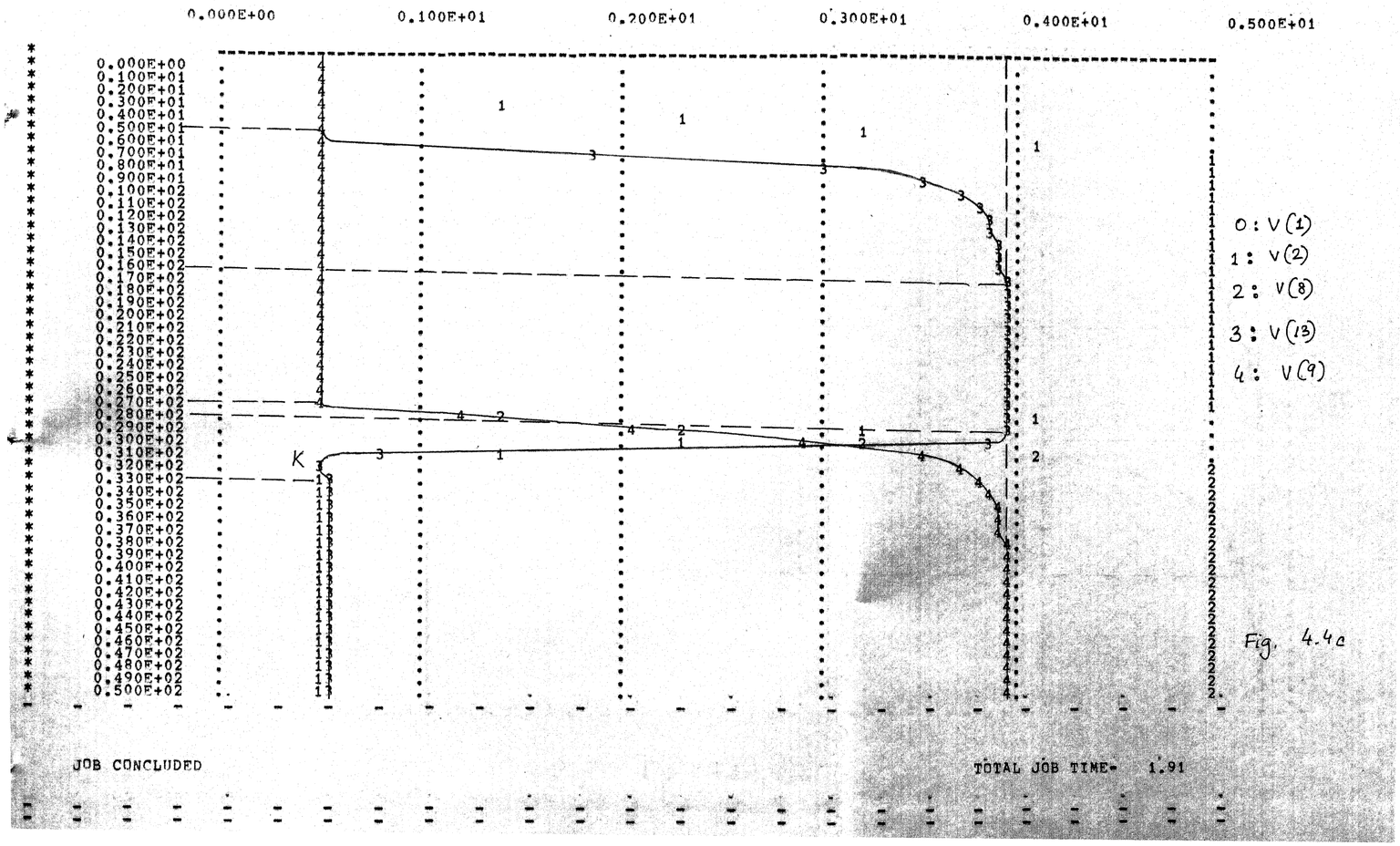


Fig. 4.4c

JOB CONCLUDED

TOTAL JOB TIME= 1.91

```
1*****************20 AUG 8 ************************  SPICE 2G.6     3/15/83 ************************14:01:20*****************
0ONE-BIT FULL ADDER WITH PASS TRANSISTOR CARRY CHAIN
0****                    TRANSIENT ANALYSIS                                    TEMPERATURE =   27.000 DEG C
0*************************************************************************************************************************

0LEGEND:
 *: V(1)
 +: V(2)
 =: V(8)
 $: V(11)
 0: V(9)
X
     TIME        V(1)
(*+=)----------   0.000D+00          2.000D+00          4.000D+00          6.000D+00          8.000D+00
($0)----------   0.000D+00          1.000D+00          2.000D+00          3.000D+00          4.000D+00
0.000D+00     5.000D-01
1.000D-09     5.000D-01
2.000D-09     5.000D-01
3.000D-09     5.000D-01
4.000D-09     5.000D-01
5.000D-09     5.000D-01
6.000D-09     5.000D-01
7.000D-09     5.000D-01
8.000D-09     5.000D-01
9.000D-09     5.000D-01
1.000D-08     5.000D-01
1.100D-08     5.000D-01
1.200D-08     5.000D-01
1.300D-08     5.000D-01
1.400D-08     5.000D-01
1.500D-08     5.000D-01
1.600D-08     5.000D-01
1.700D-08     5.000D-01
1.800D-08     5.000D-01
1.900D-08     5.000D-01
2.000D-08     5.000D-01
2.100D-08     5.000D-01
2.200D-08     5.000D-01
2.300D-08     5.000D-01
2.400D-08     5.000D-01
2.500D-08     5.000D-01
2.600D-08     5.000D-01
2.700D-08     5.000D-01
2.800D-08     1.400D+00
2.900D-08     2.300D+00
3.000D-08     3.200D+00
3.100D-08     4.100D+00
3.200D-08     5.000D+00
3.300D-08     5.000D+00
3.400D-08     5.000D+00
3.500D-08     5.000D+00
3.600D-08     5.000D+00
3.700D-08     5.000D+00
3.800D-08     5.000D+00
3.900D-08     5.000D+00
4.000D-08     5.000D+00
4.100D-08     5.000D+00
4.200D-08     5.000D+00
4.300D-08     5.000D+00
4.400D-08     5.000D+00
4.500D-08     5.000D+00
4.600D-08     5.000D+00
4.700D-08     5.000D+00
4.800D-08     5.000D+00
4.900D-08     5.000D+00
5.000D-08     5.000D+00
Y
0
0
JOB CONCLUDED
TOTAL JOB TIME            18.50
```

Fig 4.4d

```
; CIRCUIT DESCRIPTION FILE - CKT.DES
* TWO-BIT JOHNSON RING COUNTER *
.12 002
CELL
IPAST 004
006 007 001 NOELM 000 IPAST 003 INBUF 001
INBUF 001
001 002 NOELM 000 NORG2 001
IPAST 002
003 009 001 IPAST 001 IPAST 001 INBUF 001
NORG2 001
002 008 003 NOELM 000 NORG2 002 IPAST 002
IPAST 001
003 009 004 NOELM 000 NOELM 000 INBUF 002
INBUF 002
004 005 NOELM 000 NORG2 002
IPAST 003
010 007 004 NOELM 000 NOELM 000 INBUF 002
NORG2 002
005 008 010 NOELM 000 NOELM 000 IPAST 003
FINIS
GATE
NORG2 001
001 002 003 NOELM 000 NOELM 000 NOELM 000
FINIS
INPUT 001 L
001 PHAS1 NOELM 000
INPUT 002 L
002 PHAS2 NOELM 000
INPUT 003 L
003 INGEN NOELM 000
INBUF 001 L L
006 008 NOELM 000 NOELM 000
CALSB 002
CELL 006
005 NORG2 002 006 IPAST 004 007 IPAST 004 008 NORG2 001 009 IPAST 002 010 IPAST 003
005 008 002 003 001 004
007 004 002 003 001 006
CALSB 004
GATE 003
001 NORG2 001 002 NORG2 001 003 NORG2 001
004 006 009
005 007 011
005 006 010
004 007 012
CLOSE

CIRCUIT INITIALISATION FILE - INIT.CON
000
003
001 0
002 1
005
003 1
003 001 002 009 010
Y 000
000
000
000
```

Listing : 4.5a

```
* TWO-BIT JOHNSON RING COUNTER *
*********************************************

VOLTAGE WAVEFORMS AT SPECIFIED NODES:                                  ------MOSIMR------

*********************************************

TIME(N-SEC)                                VOLTAGES(VOLTS)----->

          0.000E+00     0.100E+01     0.200E+01     0.300E+01     0.400E+01     0.500E+01
```

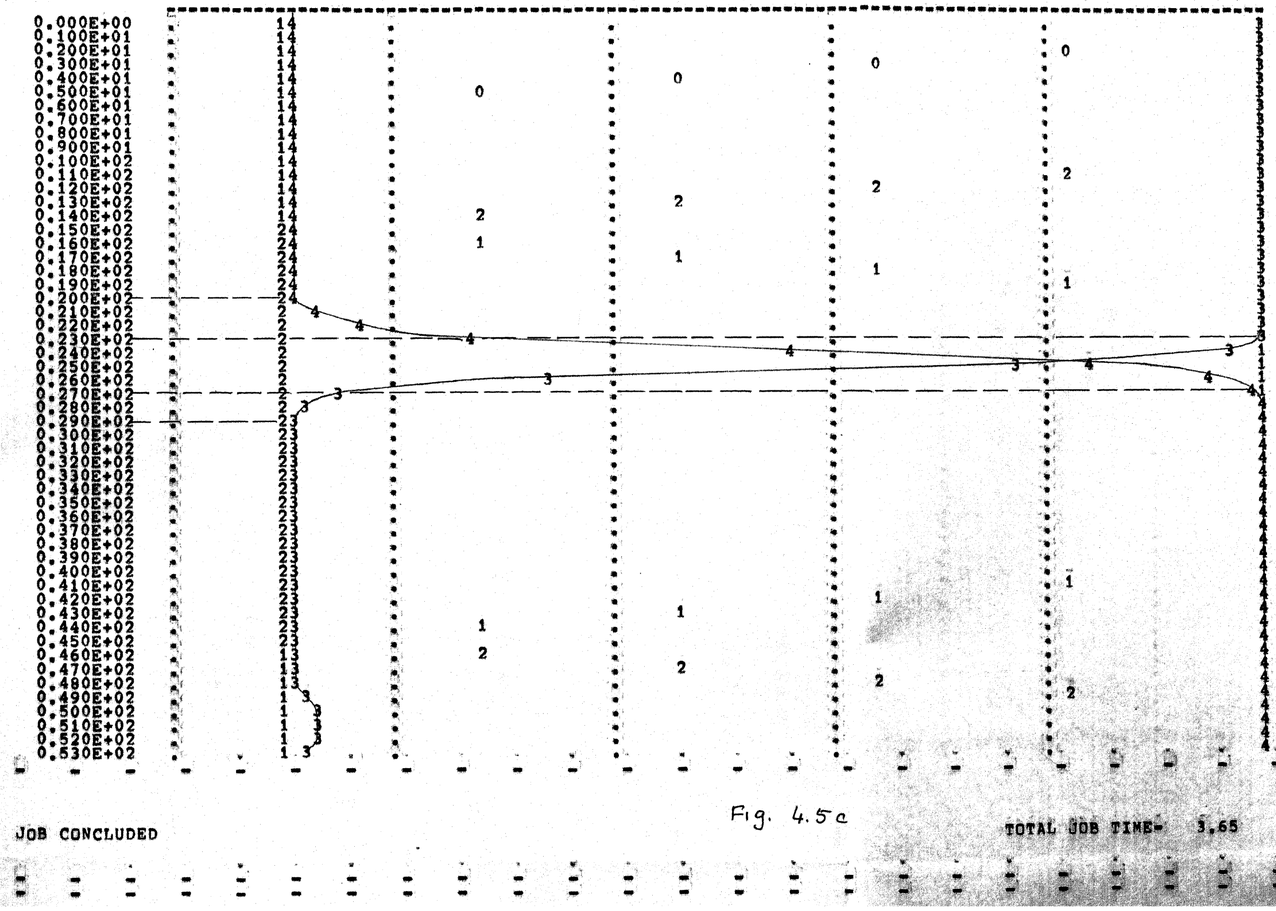
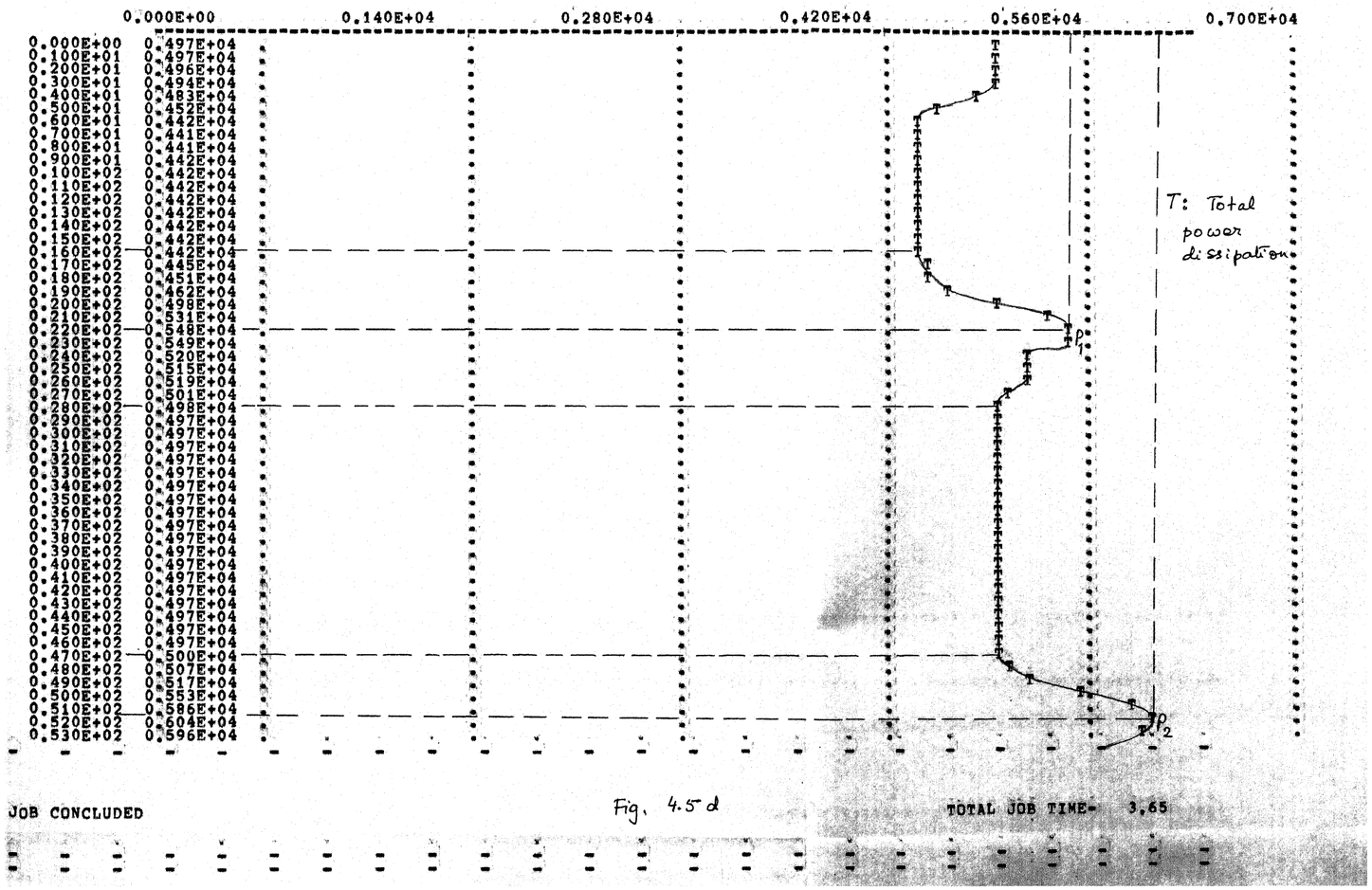


Fig. 4.5c

```
JOB CONCLUDED                                          TOTAL JOB TIME=    3.65
```

```
* TWO-BIT JOHNSON RING COUNTER *
**********************************************

TABLE II-TOTAL POWER VERSUS TIME
WAVEFORM OF TOTAL POWER DISSIPATED IN CKT. BLOCK :                    ------MOSIMR------

**********************************************

TIME(N-SEC) TOTAL POWER                          TOTAL POWER(U-WATTS)----->

                0.000E+00        0.140E+04        0.280E+04        0.420E+04        0.560E+04        0.700E+04
0.000E+00  0.497E+04
0.100E+01  0.497E+04
0.200E+01  0.496E+04
0.300E+01  0.494E+04
0.400E+01  0.483E+04
0.500E+01  0.452E+04
0.600E+01  0.442E+04
0.700E+01  0.441E+04
0.800E+01  0.441E+04
0.900E+01  0.442E+04
0.100E+02  0.442E+04
0.110E+02  0.442E+04
0.120E+02  0.442E+04
0.130E+02  0.442E+04
0.140E+02  0.442E+04
0.150E+02  0.442E+04
0.160E+02  0.442E+04
0.170E+02  0.445E+04
0.180E+02  0.451E+04
0.190E+02  0.462E+04
0.200E+02  0.498E+04
0.210E+02  0.531E+04
0.220E+02  0.548E+04
0.230E+02  0.549E+04
0.240E+02  0.520E+04
0.250E+02  0.515E+04
0.260E+02  0.519E+04
0.270E+02  0.501E+04
0.280E+02  0.498E+04
0.290E+02  0.497E+04
0.300E+02  0.497E+04
0.310E+02  0.497E+04
0.320E+02  0.497E+04
0.330E+02  0.497E+04
0.340E+02  0.497E+04
0.350E+02  0.497E+04
0.360E+02  0.497E+04
0.370E+02  0.497E+04
0.380E+02  0.497E+04
0.390E+02  0.497E+04
0.400E+02  0.497E+04
0.410E+02  0.497E+04
0.420E+02  0.497E+04
0.430E+02  0.497E+04
0.440E+02  0.497E+04
0.450E+02  0.497E+04
0.460E+02  0.497E+04
0.470E+02  0.500E+04
0.480E+02  0.507E+04
0.490E+02  0.517E+04
0.500E+02  0.553E+04
0.510E+02  0.586E+04
0.520E+02  0.604E+04
0.530E+02  0.596E+04
```




```
JOB CONCLUDED                                                    TOTAL JOB TIME=   3.65
```

Fig. 4.5 d

```
1*******23 Aug 8 ********  SPICE 2G.6    3/15/83 ********14:29:53*****
0TWO-BIT JOHNSON RING COUNTER
0****     INPUT LISTING                        TEMPERATURE =   27.000 DEG C
0******************************************************************************
```

LISTING : 4.56

```
* CKT DESCRIPTION
.MODEL DRIVER NMOS VTO=1V KP=55.2E-6
.MODEL LOAD NMOS VTO=-4V KP=13.8E-6
.MODEL PASTR NMOS VTO=1V KP=27.6E-6
.SUBCKT INBUF 1 2 4
ML1 4 3 3 0 LOAD
ML2 4 3 2 0 LOAD
MD1 3 1 0 0 DRIVER
MD2 2 1 0 0 DRIVER
C1 1 0 .04PF
C2 2 0 .02PF
C3 3 0 .04PF
.ENDS
.SUBCKT NOR 1 2 3 4
ML 4 3 3 0 LOAD
MD1 3 1 0 0 DRIVER
MD2 3 2 0 0 DRIVER
C1 1 0 .02PF
C2 2 0 .02PF
C3 3 0 .03PF
.ENDS
.SUBCKT IPAST 1 2 3
MP 1 2 3 0 PASTR
C1 1 0 .01PF
C2 2 0 .02PF
C3 3 0 .01PF
.ENDS
.SUBCKT BLOCK 1 2 3 4 5 6 11
X1 8 3 9 11 NOR
X2 6 3 5 11 NOR
X3 7 8 11 INBUF
X4 10 6 11 INBUF
X5 1 2 7 IPAST
X6 9 4 7 IPAST
X7 9 4 10 IPAST
X8 5 2 10 IPAST
.ENDS
X1 8 2 3 1 4 5 13 BLOCK
X2 4 2 3 1 6 7 13 BLOCK
X3 6 8 13 INBUF
X4 4 6 9 13 NOR
X5 5 6 10 13 NOR
X6 5 7 11 13 NOR
X7 4 7 12 13 NOR
* INPUTS
VDD 13 0 DC 5
VPH1 1 0 PULSE(5 .5 15NS 5NS 5NS 20NS 100NS)
VPH2 2 0 PULSE(.5 5 10NS 5NS 5NS 30NS 100NS)
VR 3 0 PULSE(5 .5 1NS 5NS 5NS 100NS 100NS)
* CONTROL
.TRAN 1NS 53NS
* OUTPUT REQUESTS
.PRINT TRAN V(3) V(1) V(2) V(9) V(10)
.PLOT TRAN V(3) V(1) V(2) V(9) V(10)
.END
1*****************23 Aug 8 ************************  SPICE 2G.6    3/15/83 ************************14:29:53*****************
0TWO-BIT JOHNSON RING COUNTER
0****                    MOSFET MODEL PARAMETERS                    TEMPERATURE =   27.000 DEG C
0******************************************************************************************************************************


0TYPE          DRIVER     LOAD       PASTR
                NMOS       NMOS       NMOS
0LEVEL           1.000      1.000      1.000
0VTO             1.000     -4.000      1.000
0KP          5.52D-05   1.38D-05   2.76D-05
1*****************23 Aug 8 ************************  SPICE 2G.6    3/15/83 ************************14:29:53*****************
0TWO-BIT JOHNSON RING COUNTER
0****                    INITIAL TRANSIENT SOLUTION                 TEMPERATURE =   27.000 DEG C
0******************************************************************************************************************************

 NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE     NODE   VOLTAGE
```

1*******************29 Aug 8 ************************ SPICE 2G.6 3/15/83 ************************13:32:12*****************

0TWO-BIT JOHNSON RING COUNTER
0****   TRANSIENT ANALYSIS   TEMPERATURE = 27.000 DEG C

0***

0LEGEND:

*: V(3)
+: V(1)
=: V(2)
S: V(9)
0: V(10)
X

TIME	V(3)
0.000D+00	5.000D+00
1.000D-09	5.000D+00
2.000D-09	4.100D+00
3.000D-09	3.200D+00
4.000D-09	2.300D+00
5.000D-09	1.400D+00
6.000D-09	5.000D-01
7.000D-09	5.000D-01
8.000D-09	5.000D-01
9.000D-09	5.000D-01
1.000D-08	5.000D-01
1.100D-08	5.000D-01
1.200D-08	5.000D-01
1.300D-08	5.000D-01
1.400D-08	5.000D-01
1.500D-08	5.000D-01
1.600D-08	5.000D-01
1.700D-08	5.000D-01
1.800D-08	5.000D-01
1.900D-08	5.000D-01
2.000D-08	5.000D-01
2.100D-08	5.000D-01
2.200D-08	5.000D-01
2.300D-08	5.000D-01
2.400D-08	5.000D-01
2.500D-08	5.000D-01
2.600D-08	5.000D-01
2.700D-08	5.000D-01
2.800D-08	5.000D-01
2.900D-08	5.000D-01
3.000D-08	5.000D-01
3.100D-08	5.000D-01
3.200D-08	5.000D-01
3.300D-08	5.000D-01
3.400D-08	5.000D-01
3.500D-08	5.000D-01
3.600D-08	5.000D-01
3.700D-08	5.000D-01
3.800D-08	5.000D-01
3.900D-08	5.000D-01
4.000D-08	5.000D-01
4.100D-08	5.000D-01
4.200D-08	5.000D-01
4.300D-08	5.000D-01
4.400D-08	5.000D-01
4.500D-08	5.000D-01
4.600D-08	5.000D-01
4.700D-08	5.000D-01
4.800D-08	5.000D-01
4.900D-08	5.000D-01
5.000D-08	5.000D-01
5.100D-08	5.000D-01
5.200D-08	5.000D-01
5.300D-08	5.000D-01

(*+=S0)--------------

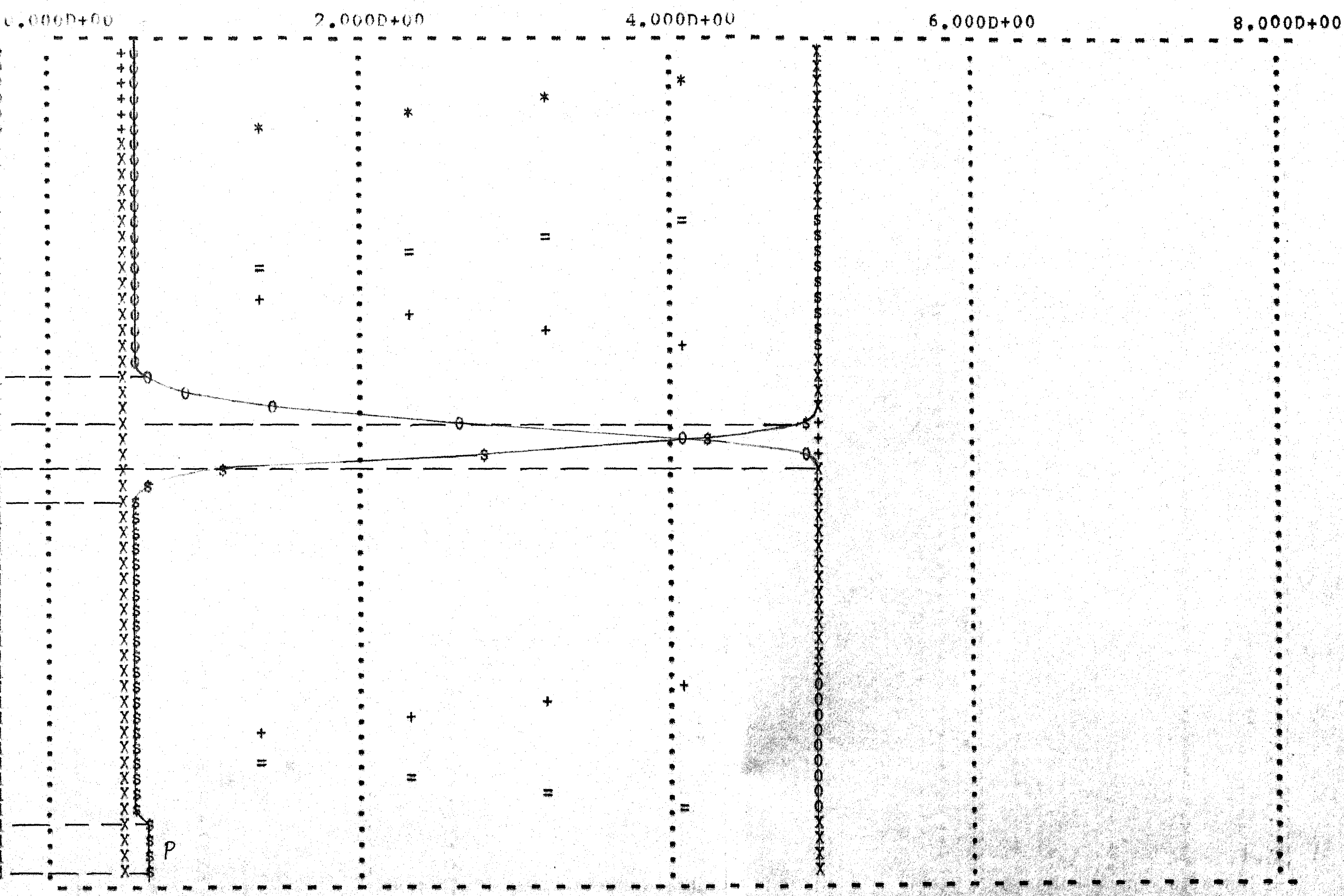
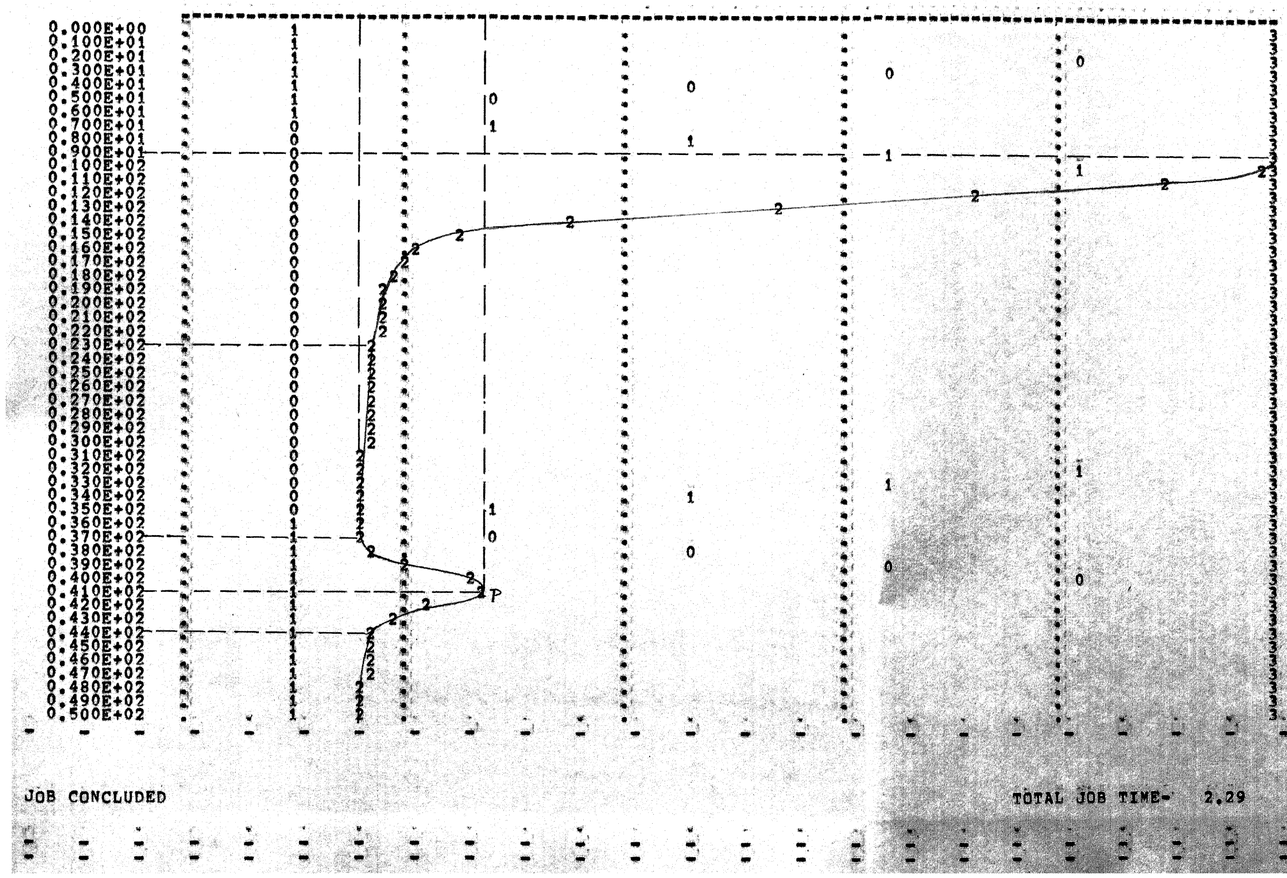


FIG. 4.5e

Y
D

JOB CONCLUDED
TOTAL JOB TIME 49.66

```
* CIRCULATING REFRESH REGISTER WITH R/W CONTROL *
*************************************************

VOLTAGE WAVEFORMS AT SPECIFIED NODES:                                  ------MOSIMR------

*********************************************

TIME(N-SEC)                                VOLTAGES(VOLTS)----->

          0.000E+00      0.100E+01      0.200E+01      0.300E+01      0.400E+01      0.500E+01
```



```
JOB CONCLUDED                                  TOTAL JOB TIME-    2.29
```

0 : V(12)

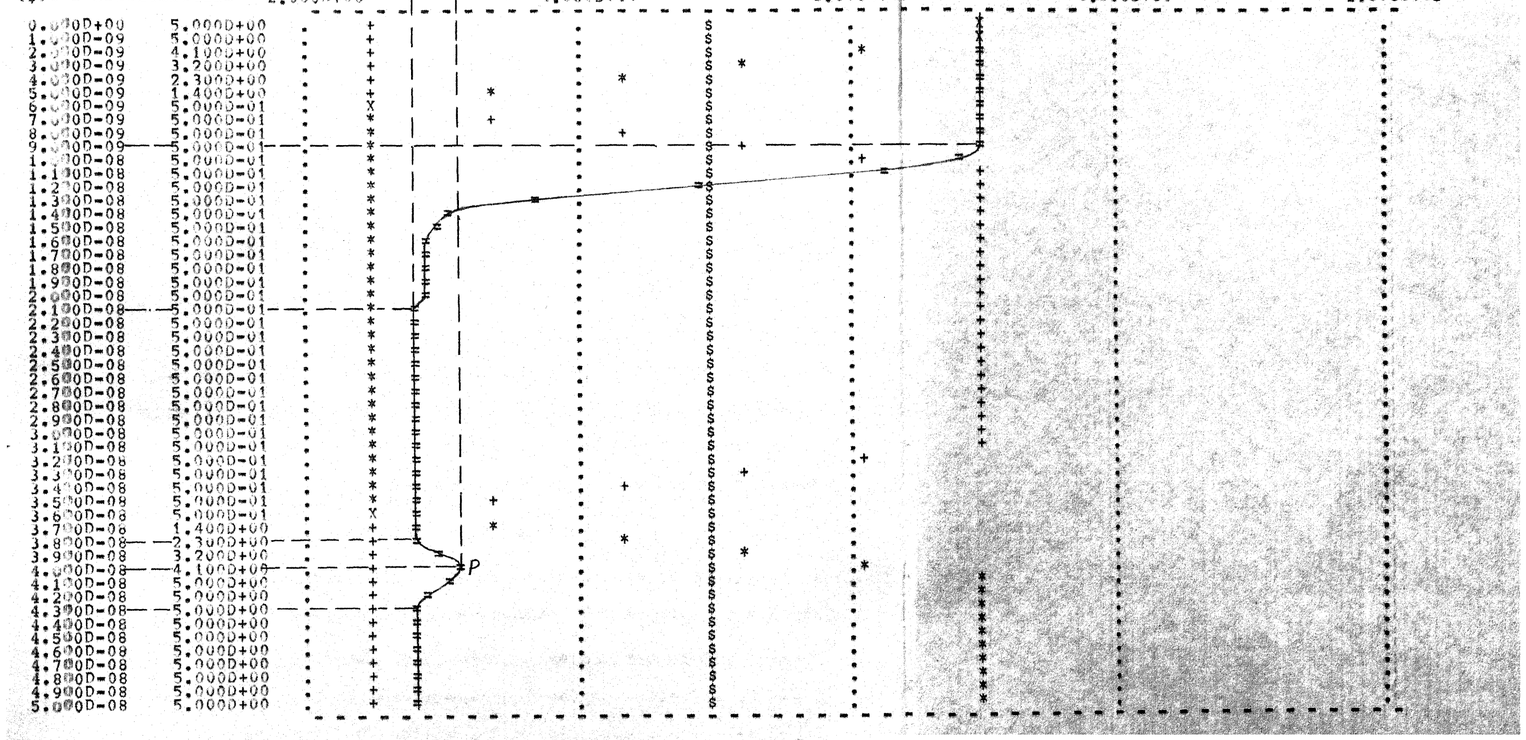
1 : V(13)

2 : V(5)

3 : V(9)

FIG. 4.6c

```
1************29 Aug 8 ********************** SPICE 2G.6    3/15/83 **********************13:41:07****************
0CIRCULATING REFRESH REGISTERS WITH R/W CONTROL
0****                    TRANSIENT ANALYSIS                                   TEMPERATURE =   27.000 DEG C
0*****************************************************************************************************************

0LEGEND:
 *: V(12)
 +: V(13)
 =: V(5)
 $: V(9)
X
     TIME         V(12)
(*+=)----------   0.000D+00          2.000D+00          4.000D+00          6.000D+00          8.000D+00
($)--+---------   2.000D+00          4.000D+00          6.000D+00          8.000D+00          1.000D+01
```




```
JOB CONCLUDED
TOTAL JOB TIME            55.24
```

Fig. 4.6 d

CHAPTER 5

CONCLUSION

5.1 ADVANTAGES ACCRUED BY MOSIMR:

MOSIMR as a special purpose program to perform time domain analysis of NMOS digital circuits has been able to gain several advantages.

5.1.1 Improved Speed of Analysis:

As apparent from the results in Chapter 4 MOSIMR proves its efficency in the analysis of comparitively large circuits. Node partitioning and effective exploitation of circuit latency are chiefly responsible for it being more than ten times faster than SPICE 2G even in the case of medium sized circuits (such as the two-bit Johnson ring counter). Single iteration of GS and NR employed by MOSIMR makes the program faster than other special purpose iterative simulators.

5.1.2 Data Storage Requirement, Minimal:

The program MOSIMR uses equational device models, thus avoiding the enormous storage required for tabular models [10].

Using nonlinear Gauss-Seidel method the program only needs to maintain a one-dimensional voltage array independent of simulation period. While RELAX [11], [12], using waveform

relaxation method, needs to store complete waveforms of N circuit nodes over a whole simulation period (consisting of T time steps) in a TxN array. Thus memory requirement in waveform relaxation method is proportional to the period of simulation. Estimated storage needed in MOSIMR for a large circuit with 1000 nodes, is 55K words of core, independent of simulation period. This may be compared with the data space requirement of 2.4M bytes by RELAX to analyse an MOS circuit with 1000 nodes, over 100 time steps [11].

5.1.3 Multifarious Applications of MOSIMR:

Though a special purpose simulator, different features of MOSIMR can be utilised to study various circuit aspects as follows:

i) MOSIMR computes the initial condition of the circuit under steady state, based on a full-fledged logic simulation. Hence it can potentially be used to simulate logic of a digital NMOS circuit under any steady state condition. Logic function of a complex combinatorial circuit can thus be found out with the help of MOSIMR applying different input combinations. Along side, quescent power dissipation of a circuit under any steady state condition can also be obtained.

ii) Speed and power dissipation becomes important factors in the design of sequential circuits and finite state machine

structures. The usual timing analysis feature of MOSIMR can be used to study response of such circuits with clocking. This way the maximum allowable clock frequency can be found out for a design.

iii) The external definition of input (EXDEF) allows arbitrary waveforms to be applied to the circuit. Taking advantage of this feature a large circuit can be broken up into blocks which can be analysed independently, over the entire simulation period. The output from the previous block is printed in the output file OUT.NXT. This output goes as EXDEF input to the next block and OUT.NXT is renamed INP.PRE to act as an input file during the analysis of next block. Data, read-in from INP.PRE, in every simulation step, defines the EXDEF input of the next block.

iv) MOSIMR can detect the signal-wave profile, as it travels through a large circuit. Hence a novel application of MOSIMR would be as a signal tracer in a large circuit.

v) Interactive user control over the simulator allows the user at the end of a specified simulation period to decide whether to STOP, RESTART or CONTINUE further simulation. This feature can be used in dividing a long simulation period into subintervals. As simulation is completed smoothly in one subinterval, the user can direct the simulator to continue to the next.

5.2 LIMITATIONS IN MOSIMR AND SUGGESTIONS FOR BETTERMENT:

Circuit initialisation related with the 'On' pass transistor makes an approximation in computing the source output voltage, when the drain input voltage is unknown and the input is at logic '0' (Sec. 3.3.2). This may be avoided by conducting the voltage initialisation trace along the signal flow path, as done in logic simulation.

Handling of floating capacitance in MOSIMR puts a constraint over the time step h (Sec. 2.2.1). Better way to handle these would be as done in RELAX [11].

In MOSIMR the link graph structure of the circuit is built up with the help of linked list. Implementation of the linked list data structure would have been easier with PASCAL. Also recursion in PASCAL would support multiple-level subcircuit nesting.

Alternative to the linked list representation of the directed circuit graph, is the dependency matrix approach (Sec. A.3.11.3). In this method the program can be easily adapted to simulate circuit in transistor level and allow definition and use of complex logic gates.

In MOSIMR accuracy and stability are jeopardized with large value of simulation time step h. If stability is our

concern a method should be formulated to redefine GG (Sec.2.4) in a way such that single iteration of Newton Raphson (NR) gives an asymptotically correct result, with $h \rightarrow \infty$. On the other hand to ensure accuracy NR iterations must be carried to convergence in each Gauss Seidel (GS) relaxation. Also several iterations of GS relaxation must be performed per time step, to gain at least partial convergence of solution, in the analysis of non-unidirectional circuits.

5.3 SCOPE FOR FUTURE WORK:

MOSIMR has been built to simulate NMOS circuits with basic ratioed logic gates. Further effort can be directed in expanding the simulator for ratioless gates, complex gates and finally for CMOS circuits.

This will involve mainly incorporation of transistor level simulation (over and above basic gate-level simulation), introduction of new device models and definition of new circuit elements.

APPENDIX A

ELUCIDATION OF FEW TERMS AND CONCEPTS

Here, some terms and concepts, often used and referred in the preceding chapters, have been elucidated with definition, brief explanation/derivation and illustrative examples.

A.1 OVERHEADS:

By 'overheads' is meant here the extra processing (not directly connected with bare circuit analysis), which is required in order to make a program general purpose and user oriented.

It may imply extra CPU time as well as extra storage for simulation.

i) A language translation stage at the beginning of a simulator program to allow the user to specify input data in free format;

ii) Storage of models of a large variety of devices to make the simulator generally useful;

iii) Extra accuracy in device models, than is often required;

iv) Elaborate outputting - giving details in listing, in which the user is often not directly interested;

: are, to name a few, which principally cause overheads in a general purpose circuit simulator.

A.2 MACROMODEL:

A circuit macromodel is a unit within a circuit which is actually a physical conglomerate of several elements and devices. So it is rather a complex element which can be modeled either on the basis of component element models (i.e. device level macro-modeling) or simply by characterising the input/output functional relationship without regard to the physical interior of the unit (i.e. functional macromodeling).

In MOSIMR a macromodel is termed, a circuit element.

A.3 RELAXATION METHOD:

General scheme of solving a system of algebraic differential equations by relaxation method [11], [12] consists of two major processes, namely the assignment partitioning process and the relaxation-iteration process.

$$F(\dot{y},y,u) = 0 \qquad \text{(A.1a)}$$

$$E(y(o)-y_o) = 0 \qquad \text{(A.1b)}$$

(A.1a) is the system of equations to be solved with (A.1b) as initial conditions, where

$y(t) \in R^p$ is the vector of p unknown variables at time t.

$\dot{y}(t) \in R^p$ is the time derivative of y at time t

and $u(t) \in R^r$ is the vector of r inputs at time t

A.3.1 Assignment Partitioning Process:

In assignment partitioning process, each unknown variable is assigned to an equation of (A.1a) in which it is involved. Thus (A.1a) is partitioned into p disjoint subsystems:

$$\begin{aligned} F_1(\dot{y}_1, y_1, d_1, u) &= 0 \\ F_2(\dot{y}_2, y_2, d_2, u) &= 0 \qquad \text{(A.2a)} \\ &\vdots \\ F_p(\dot{y}_p, y_p, d_p, u) &= 0 \end{aligned}$$

$$E(y(o) - y_o) = 0 \qquad \text{(A.2b)}$$

where $y_i(t)$ is the variable to be solved from

$$F_i = 0 \qquad \text{(A.2c)}$$

with $d_i \subseteq \{y_1, y_2, \cdots y_{i-1}, y_{i+1}, \cdots y_p\}$

Consisting of other variables present in eqn. (A.2c) which are considered as inputs to the subsystem, represented by eqn. (A.2c), with their trial values.

Consideration of d_i helps to decouple the eqn. (A.1a) into those in eqn. (A.2a) and hence the variables included in d_i are called the decoupling inputs of eqn. (A.2c).

A.3.2 Relaxation-iteration Process:

The two most commonly used types of relaxation methods namely the Gauss Seidel (GS) and the Gauss Jacobi (GJ) are to be considered here.

The initial trial solution for y(t) is a guess. During each iteration each decomposed subsystem of eqn. (A.2a) is solved for its assigned variable using the trial values for the variable and the corresponding decoupling inputs.

For the GS relaxation, the solution obtained for a variable by solving a decomposed subsystem is immediately used to update the trial values - so that it can be used in the solution of other decomposed subsystems which follow in the same iteration step.

For the GJ relaxation, all trial values are updated only at the beginning of next iteration.

The relaxation process is carried out repeatedly as the trial values converge to the actual solution.

The convergence criteria for relaxation-iteration are given in [11] and these require diagonal dominance of the Jacobian of the discretised equations obtained from eqn. (A.1a).

A.3.3 GS/GJ Time Advancement Formula:

This is a recursive formula for solving (A.1a) in successive time steps with only one relaxation iteration performed per time step. As this is a kind of numerical integration the method is called Guass-Seidel/Gauss-Jacobi Integration algorithms, according as the relaxation scheme used is GS or GJ.

Illustrating the time advancement scheme for a linear system described by linear state eqns.:

$$\dot{x} = Ax \qquad \text{(A.3a)}$$

and $$x(o) = x \qquad \text{(A.3b)}$$

where $x(t) \in R^n$ the state vector to be solved at time t consisting of n variables.

$A \in R^{nxn}$ the coefficient matrix = L+D+U

where L is strictly lower triangular

D is diagonal

and U is strictly upper triangular

The G.S. integration formula is -

$$[\, I - h(D+L)\,]\, x_{t+1} = [I + hU]\, x_t \qquad \text{(A.4a)}$$

and the G-J integration formula is -

$$[I-hD]\, x_{t+1} = [I+h(L+U)]\, x_t \qquad \text{(A.4b)}$$

where I is the identity matrix,

x_{t+1} is the state solution at t+1,

x_t is the trial solution, which rather being solution of the previous iteration is the solution of previous time step (as only one iteration of G-S/G-J relaxation is performed per time step).

A.3.4 Network Ordering and its Effect on Timing Errors:

In G-S/G-J integration algorithm (Sec. A.3.3) only one iteration of relaxation is performed. Now we should recognise a fact here and that is circuits with unidirectional properties can be analysed exactly by performing only one iteration of G-S relaxation if the computation of the node voltages are ordered in such a way that the decoupling inputs to each equation of the partitioned system are evaluated. for the present time step before the particular equation is solved.

This ordering of computation in the case of an unidirectional circuit, for which a link graph can be built (A.4.4) (where computation of a node depends on the higher order nodes alone and not on the lower order nodes (A.3.9.3)) is naturally according to the precedence order of signal flow.

Thus network ordering has a profound effect on the rate of convergence of G-S method. However,ordering of computation

does not have any effect on G-J method where the decoupling inputs retain the same values throughout the whole iteration step.

Now timing error is a phenomenon that is always evident in G-J method and becomes evident in G-S case when computations are done in an arbitrary order or even with network ordering when the circuit is not unidirectional.

Fig. A.1a: Unidirectional circuit

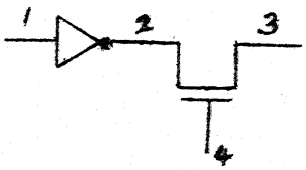


Fig. A.1c: Circuit with bidirectional element

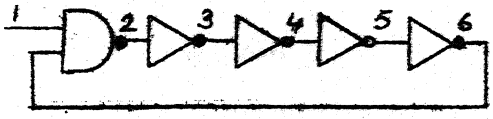


Fig. A.1b: Circuit with feedback

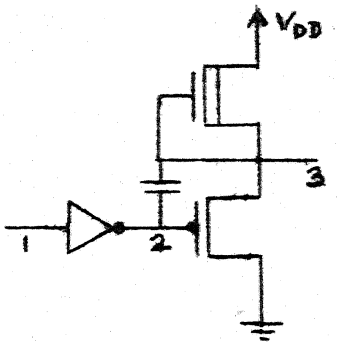


Fig. A.1d: Circuit with floating capacitor

Considering the example of unidirectional circuit in Fig. A.1a the ordered G-S would proceed to compute the nodes in the sequence 1-2-3-4 following the flow of signal in the circuit.

at 6 at t_k and a timing error of one time step is incurred.

Similarly computation of 2 at t_{k+1} depends on 3 at t_k in Fig. A.1c and Fig. A.1d, incurring timing errors of 1 simulation step in each case.

A.3.5 Latency:

For most circuits the fraction of nodes which change their voltage values at a given point in time, decreases as the circuit size increases. For circuits containing over 500 MOSFET's, fewer than 20% [12] of the node voltages change (significantly) over a simulation step. This inactivity in circuits have been termed 'latency'.

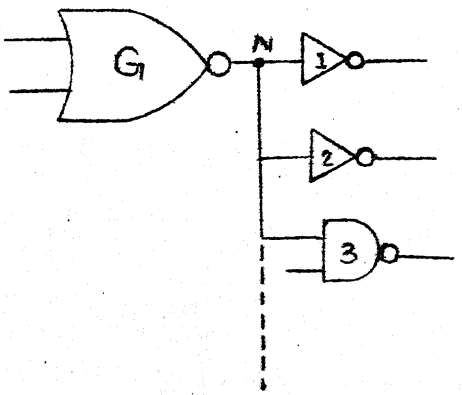
When circuit analysis is scheduled in such a way that at any time only the active nodes are evaluated, leaving out the redundant calculations of the inactive parts, we call it 'exploitation of latency'.

A.3.6 Leading Element:

In the linked structure based on the concept of observability (Sec. A.3.9), node N in Fig. A.2 is observed by all the three elements 1,2,3 which together form a chain where 2 may be thought of as the next brother (NB) element of 1; 3, the NB of 2 and so on. The first element 1 which

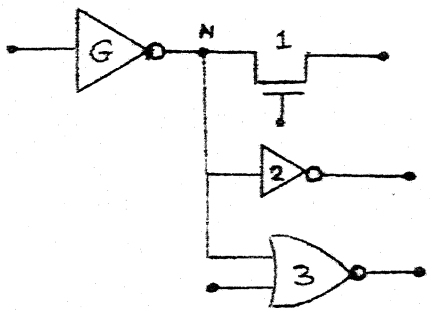
is recognised as the successor (succ.) element at N,is termed the leading element of the NB chain from which a trace scheduler can visit the subsequent NB's following the NB pointers.

Similarly in the fan-out table representation of dependency, the first element, in the list of fan-out elements, recorded for node N, is the leading element.



Leading element in the chain of next brother elements, following gate G at node N, is the one marked '1'

Fig. A.2.a



Leading element among the fan-out elements at N,is the one topping the fan-out list, namely the pass transistor marked '1' here.

Fig. A.2.b

A.3.7 Selective Trace:

Whenever the voltage at a node changes, it is possible to schedule all the elements observing the node to be processed.

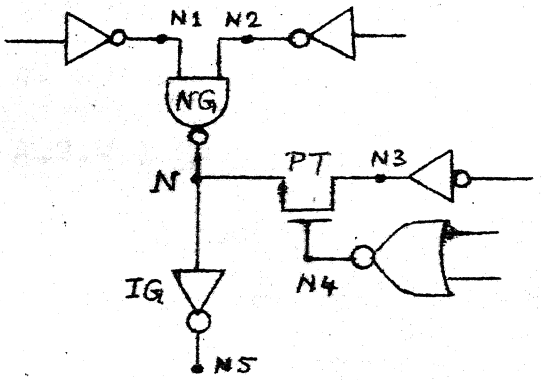
In this way the effect of a change at the input of a circuit may be traced as it propagates to other circuit nodes via a linked list/fan-out table etc.

Since the only nodes processed in this way are those which are affected directly by the change, this circuit trace procedure is selective and is known as 'selective trace'.

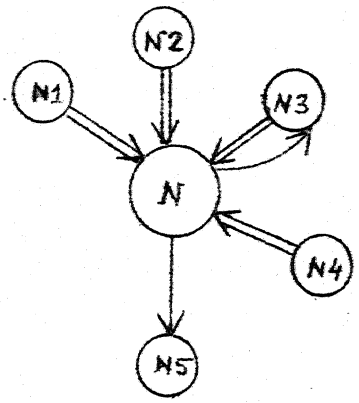
A.3.8 Scheduler:

A scheduler is a program which plans which nodes are to be computed and in what sequence in a particular time step of simulation.

A.3.9 Controllability and Observability of Nodes:



A piece of network showing the node of interest N and its neighbouring nodes N1-N5 which have controllability and/or observability relations with N.



⟹ denotes control relation.

⟶ denotes observability relation.

Directed graph representation of the piece of network.

Voltage at the test node N in Fig. A.3a can be changed by varying the voltage(s) at any or more of the nodes N1, N2, N3 and N4, whereas N5 does not have any effect on N. So we say N1, N2, N3 and N4 are the nodes which control N.

On the other hand any change of voltage at node N cannot be felt at N1, N2 and N4 but it can surely be felt at N3 and N5. So N3 and N5 are said to be the nodes which observe N.

These relations are conveniently represented by a directed graph where the nodes of the graph correspond to the circuit nodes and the edges of the graph correspond to the controllability and/or observability relations between nodes through elements. The directed graph-equivalent of the piece of network in Fig. A.3a is given in Fig. A.3b.

A.3.9.1 Reachability:

A node j is reachable (by controllability/observability relation) from node i if and only if there is at least one directed edge from i to j [18].

A.3.9.2 Strongly connected graph:

A graph is strongly connected if and only if every node is reachable (Sec. A.3.9.1) from any other node. This means that in a strongly connected graph, every node is in at least one loop [18].

A.3.9.3 Maximal strongly connected subgraph:

A maximal strongly connected (MSC) subgraph is a set of all possible nodes that are strongly connected (Sec. A.3.9.2) with each other, in a graph.

This designates the maximum set of nodes that are involved in conflict (Sec. A.3.9.4) from controllability and observability view point [9].

A.3.9.4 Order of nodes

Higher order nodes are those evaluated earlier than the lower order nodes [9].

A.3.9.5 Conflict:

Conflict is said to exist when computation of a higher order node depends on any lower order node(s) [9].

A.3.9.6 Link graph:

A link graph is one that contains no strongly connected (Sec. A.3.9.2) or unconnected subgraphs in it. It is therefore an acyclic digraph. (i.e. loop free) and the nodes in it can be arranged in an ordered list. Physically this corresponds to a unidirectional circuit where in principle, with ordering of elements, nodes can be computed without any conflict (Sec. A.3.9.5).

A.3.10 Resolution of MSC'S:

MSC's (Sec. A.3.9.3) which cause conflict due to presence of loops, must be resolved by breaking the loops (known as decyclization in graph theory) and introducing additional control and/or monitor points associated with each broken edge - to maintain equivalent of the derived graph.

In fact all edges entering the node of conflict are broken this way and additional control and/or monitor points introduced.

The node of conflict 'N' is hereby partitioned and a subsystem is formed which includes the node in question and all the control points introduced. In such a subsystem node N can be evaluated independently with the help of higher order nodes and the additional control points introduced in the subsystem block.

Hence node partitioning can resolve MSC's and allow a link graph structure (i.e. an acyclic connection of the above subsystems) to be formed.

A.3.11 Node Partitioning and Ordering:

Node partitioning and ordering can be done wholly on the basis of controllability and observability relations respectively, of nodes.

A subsystem obtained after partition should contain the node of interest and all the controls on that node i.e. all the fan-in elements (Sec. A.3.11.1).

Ordering of partitioned subsystems determines the order in which node(s) to be visited after computing a node i. Naturally those nodes which observe node i are the ones to be visited after updating node i.

A linked list [19] implementation of the ordering is possible where the pointers depict observation edges that lead to the next node(s) to be visited.

A fan-out list for each node is an alternative option (Sec. A.3.11.2).

In Fig. A.3a for example the node of conflict N, when partitioned, gives a subsystem that includes the node in question i.e. N, the fan-in elements at N i.e. the NAND gate (NG) and the pass transistor (PT) and fan-out elements at N i.e. the pass transistor (PT) and the inverter (IG). The fan-in, fan-out information for each node can be kept in tables or other wise can be elegently represented with the help of dependency matrix (Sec. A.3.11.3).

A.3.11.1 Fan-in Element:

Fan-in elements of node i are defined as those which play some part in determining the voltage at node i (i.e.

those elements which cause some entries of column i of dependency matrix to be 1).

A.3.11.2 Fan-out Element:

Fan-out elements of node i are the ones whose operating conditions are directly influenced by the voltage at node i (i.e. those elements which cause some entries of row i of dependency matrix to be 1).

A.3.11.3 Dependency matrix:

A signal flow graph which determines the network ordering for the processing of nodes, can be generated by means of dependency matrix. This is the adjacency matrix, of the circuit digraph, of order nxn; where n = total number of nodes.

Entry P_{ij} in the matrix = 1 if there is a directed edge that joins node x_i to node x_j. Otherwise $P_{ij} = 0$. When $P_{ij} = 1$ the physical implication is, node voltage v_i influences node voltage v_j via a device equation. That is x_i is a fan-in node of x_j and conversely x_j is a fan-out node of x_i.

APPENDIX B

MOSIMR : USER'S GUIDE

MOSIMR is a special purpose circuit simulator for time domain analysis of digital NMOS circuits. This manual elaborates how to interact with the simulator to get desired results. It also points out the constraints and idiosyncrasies of the program in its present version.

B.1. THE INPUT/OUTPUT FILES:

Input data is read from three files, namely:

1) CKT.DES - This contains the linked description of the circuit to be analysed, in terms of elements (table B.1) and subcircuits.

2) INIT.CON - This contains the initial condition of the circuit and the output requests.

3) INP.PRE - This contains the EXDEF(table B.2) inputs in a specific order, tabulated for every simulation time step, after t=0, over the entire simulation period.

Results are printed in the following output files:

1) VTABL.DAT - This contains the requested node voltages printed against time over the whole simulation

period. Default printing/plotting step is around 1 nsec. At most 10 voltages can be requested to be printed and plotted.

2) VCURV.DAT - This contains the plot of requested node voltages against time. Numerals Ø-9 are used to represent the (1-10) consecutive requested node voltages.

3) PTABL.DAT - This contains the requested elem. powers printed against time over the entire simulation period. Default printing/plotting step is around 1 nsec. At most 10 elements can be requested for their powers to be printed and plotted.

4) PCURV.DAT - This contains the plot of element-power curves. Numerals Ø-9 are used to represent the powers of the consecutive (1-10) requested elements.

5) TPCURV.DAT - This contains the print and plot of the total power dissipation of the circuit, over the entire simulation period. The letter 'T' is used to plot the power waveform. Default printing and plotting step is around 1nsec.

B.2 INPUT FILE FORMATS:

The data provided in the input files CKT.DES, INIT.CON and INP.PRE, are in alphanumeric, integer or real modes, with following formats.

B.2.1 General Formatting of Data in CKT.DES:

Each line of data constitutes a data card of the input deck. The sequence of data cards in CKT.DES is as follows;

1) Title card - This contains an alphanumeric string of maximum 125 characters.
FORMAT (3X, 25A5)

2) MAXNOD, NSUB - This contains the total number of global nodes in the circuit description (MAXNOD) and the no. of subcircuits (NSUB) described.
FORMAT(3X, 2(I3,1X)

3) Subcircuit name - This contains the name of the subcircuit to be described.
FORMAT (3X, A5)
Skip this card and the subsequent subckt. description cards (upto FINIS) if NSUB is specified 'Ø' in the previous card.

4) NAME, NO - This contains the name (NAME) and number (local) (NO) of the subckt. elements being specified.

FORMAT (3X, A5, 1X, I3)

5) Subckt. element specification - This contains the element specifications in the following sequence:

Input node(s) (local), output node(s) (local) name and no. (local) of next brother (NB) element(s), name and no. of successor (succ.) element(s).

The exact format in which the above data are specified depends on the type of the elem. and can be obtained from Table B.1.

After this elem. specification the user specifies all the remaining subckt. elements in the same way, using cards 4 and 5.

6) FINIS - This contains the reserved word 'FINIS', indicating the end of subckt.description.

FORMAT (3X, A5)

After this the user describes the remaining subckts. (if NSUB > 1) in the same way, using cards 3-6.

After all the subckts. have been described the main circuit elements are specified using cards 7

7) NAME, NO, LIN(I), I = 1,5 - This card contains the name (NAME) and number (global) (NO) a main ckt. elem. / corresponding to the input and output nodes of the element there is a variable (LIN) to be specified. Specify LIN as 'L' in case the element is to be linked with a subckt. elem. at the corresponding I/O node. Otherwise specify LIN as 'N'.

FORMAT(3X, A5, 1X, I3, 1X, 5(A1,1X))

8) Main ckt. elem. specification - This contains the element specifications in the following sequence; Input node(s) (global), output node(s) (global), name and number (global) of the NB element(s), name and number (global) of the succ. element(s).

Exact format depends on the elem. type and is given in Table B.1.

After this, the user specifies the remaining main circuit elements and inputs, in the same way, using cards 7 and 8.

9) Subcircuit call cards - Calls to the previously described subcircuits are placed after all the main circuit elements and inputs are specified.

Cards 9(a) - 9(d) are used to place repeated calls to a particular subckt.

9a) CALSB, NO - This card contains the reserved word 'CALSB' indicating subckt. call, and the number of calls (NO) to the same subckt. FORMAT (3X, A5, 1X, I3). Skip the cards 9(a) - 9(d) if NSUB in card 2 is 0.

9b) NAMSUB, NOEX - This card contains the name of the subckt. (NAMSUB) and the no. of external nodes of the subckt. (NOEX).

FORMAT (3X, A5, 1X, I3)

9c) ((EXTERN(I,J), J=1,3), I=1, NOEX) - This card contains dummy external nodes (EXTERN(I,1)) of the subckt. and names (EXTERN(I,2)) and local nos. (EXTERN(I,3)) of the subckt. element to be linked with the main ckt. environment, at the dummy ext. node.

FORMAT(3X, 9(I3,1X, A5, 1X, I3, 1X))

9d) (EXTNOD (J), J=1,NOEX) - This card contains the set of actual nodes that replace the dummy external nodes of the subckt. The actual nodes are listed in the same order as the dummy nodes in card 9(c)

FORMAT(3X, 9 (I3, 1X))

If No. in card 9(a), is greater than 1, then the same subckt. is called more than once in different main circuit environments. For the remaining cases the set of actual nodes are listed in the same way, after this, using data card 9(d).

After all 'NO' sets of actual nodes have been given like this, calls to the remaining subckts (if NSUB in card 2 is > 1) are placed in the same way using cards 9(a) - 9(d).

10) CLOSE - This card comes after all the subckt. calls have been placed. It contains the reserved word 'CLOSE', indicating the end of the circuit description file, CKT.DES.

FORMAT (3X, A5)

B.2.2 General Formatting of Data in INIT.CON:

1) NNODE, WM, WP - This card contains the no. of nodes for which path-length stray capacitance would be computed (NNODE).

Alongside, the widths of the metal lines (WM) and poly lines (WP) are also mentioned, if these are different from default values.

Default value for WM = 9.0 μm

and for WP = 6.0 μm

FORMAT(3X, I3, 2(1X, F3.1))

2) NODE, NTYP, PATHL - This card contains the no. of the nodes whose capacitance is to be computed (NODE), type of the node (NTYP) which is either 'M' or 'P', indicating metal or polysilicon respectively and the path length (PATHL) in μm. (PATHL cannot exceed 1000 μm).

If PATHL is given as zero the default value assumed is 30 μm.

If NTYP is left a blank, default type assumed is 'M'.

Thus default value of stray capacitance of any node (unspecified) is .0081 pF.

FORMAT(3X, I3, 1X, A1, 1X, F4.1

Skip this card and go to card 3 if NNODE in card 1 is equal to zero.

If NNODE > 1 the remaining nodes are specified in the same way, after this, using card 2.

3) NO - This card contains the no. of nodes whose initial logic states are specified (NO).

FORMAT (3X, I3)

4) NODE, LOGVAL - This card contains the node no. (NODE) and its specified logic state (LOGVAL) (either a '0' or '1').

2) NODE, NTYP, PATHL - This card contains the no. of the nodes whose capacitance is to be computed (NODE), type of the node (NTYP) which is either 'M' or 'P', indicating metal or polysilicon respectively and the path length (PATHL) in μm. (PATHL cannot exceed 1000 μm).

If PATHL is given as zero the default value assumed is 30 μm.

If NTYP is left a blank, default type assumed is 'M'.

Thus default value of stray capacitance of any node (unspecified) is .0081 pF.

FORMAT(3X, I3, 1X, A1, 1X, F4.1

Skip this card and go to card 3 if NNODE in card 1 is equal to zero.

If NNODE > 1 the remaining nodes are specified in the same way, after this, using card 2.

3) NO - This card contains the no. of nodes whose initial logic states are specified (NO).

FORMAT (3X, I3)

4) NODE, LOGVAL - This card contains the node no. (NODE) and its specified logic state (LOGVAL) (either a '0' or '1').

FORMAT (3X, I3, 1X, I1)

Skip this card if NO in card 3 is equal to zero.

If NO > 1 the remaining nodes are specified in the same way, after this, using card 4.

5) NODINT - This card contains the no. of nodes of interest (NODINT) whose voltages are to be printed and plotted against time. The maximum value allowed for NODINT is 10.

FORMAT less.

6) (IVNODE (I), I = 1, NODINT) - This card contains the nodes of interest as a request list (IVNODE). IVNODE cannot include any node that belongs to a subckt.

FORMAT less.

7) NTPWR, NEPWR - This card contains the variable NTPWR, which can be either a 'Y' or an 'N', and the total no. of elements whose powers are requested to be printed and plotted (NEPWR).

If NTPWR = 'Y' the total power dissipation of the circuit is computed and printed/plotted. If NTPWR = 'N' total power dissipation is not calculated. Maximum value of NEPWR, allowed is 10.

FORMAT (3X, A1, 1X, I3)

8) (IPELM(I,J), I=1,2), J=1, NEPWR) - This card contains the name (IPELM) (1,J)) and no. (IPELM(2,J)) of the elements whose powers are to be printed/plotted. The request list must not contain any element included in a subckt. Skip this card, if NEPWR in card 7, is zero.

FORMAT(3X, 10(A5, 1X, I3, 1X)

9) NNXT - This card contains the total no. of nodes (NNXT) whose voltages are to be printed in every simulation step in the file OUT.NXT.

FORMAT less

10) (NEXT(I), I=1,NNXT) - This card contains the nos. of the nodes whose voltages are to be printed in OUT.NXT. There must not be > 15 such nodes in the request list.

FORMAT less.

Skip this card if NNXT, in previous card is equal to zero.

11) INPRE - This card contains the total no. (INPRE) of EXDEF (table B.2) inputs.

FORMAT less.

12) (NPREV(I), I=1,INPRE) - This card contains the nos. of those nodes, where EXDEF inputs are applied

FORMAT less.

13) PMAX, PMAXT - This card contains the maximum limits for the power scales in the plots for i) element power dissipation (PMAX) and ii) total power dissipation (PMAXT).

FORMAT less

This marks the end of data inputs through files.

B.3 Interaction through TTY:

When the simulator is run with above data files the user may be asked to provide some more data interactively through TTY as follows;

1) If after circuit initialisation, some nodes are found floating, (such as output, terminal of an 'off' pass transistor), the user is asked to assign a logic state '0' or '1' to these nodes to his liking.

2) Before voltage/current initialisation, the program accepts few NMOS device parameters from the user, through TTY. Any parameter typed as '0', is assigned its default value.

Parameter	Default value
VDD (power supply voltage)	5V
MU (surface mobility of electrons)	800 cm^2/V-sec.
EPS (dielectric constant of SiO_2)	0.345×10^{-6} μF/cm
TOX (oxide-thickness)	1.0×10^{-5} cm

Parameter	Default value
RATIOD (W/L ratio of driver transistor)	2
RATIOL (W/L ratio of load trans.)	0.5
RATIOP (W/L ratio of pass trans.)	1
RATN2 (W/L ratio of load trans. in a two-input NAND gate)	.25
RATN4 (W/L ratio of load trans. in a four-input NAND gate)	.125
CGS (Gate to source capacitance of an NMOS transistor)	.01 pF
CGD (Gate to drain capacitance of an NMOS trans.)	.01 pF

3) Specifications for the INGEN, CLOCK and PHAS1/PHAS2 inputs (Table B.2) are accepted through TTY, from the user, just before the analysis of the initialised circuit.

First the nonclock inputs are taken up one by one and the user is asked to specify the following for each input;

a) The input-type (Table B.2)

b) Timings for each zone (rise/fall edge, high/low period) of the input waveform (if the input type ≠ 'CONST').

All timings are in the form NXDELTA, where N is a positive integer and DELTA is the simulation time step. To specify a zone timing, the user simply types in the value of N. For a

For a rising/falling edge the rise/fall time can not be specified less than 4xDELTA.

After all the INGEN inputs have been specified, the simulator accepts the timings of the clock inputs, namely 'CLOCK' and PHAS1/PHAS2, in the same way as timings for non 'CONST' INGEN inputs are accepted.

Finally the user is asked to specify the simulation period, which is also of the form NXDELTA.

At the end of the simulation period, the user is asked to guide the future course of action of the simulation. At this point,

i) Typing an 'S', stops the simulator

ii) Typing an 'R', restarts simulation

iii) Typing a 'C', continues simulation

On RESTART'ing, the circuit is reinitialised and the user has to respecify all data, which are accepted by the simulator, through TTY.

On CONTINUE'ing, circuit simulation is carried on, with the currently existing state of the circuit as initial state. Before starting analysis, the user is asked to type-in the new simulation period, 'NPER' and waveform specifications of inputs, if any. The simulator asks the user whether there is any

input to be specified in the new period. If there is any such, the user types-in the 'input no.' and subsequently gives the waveform specifications, for that input, in the same way as told in 3(b). Whereas, if there is no (more) input to specify, the user types-in a 0 and the simulator starts analysis over the new simulation period, forth with. Note, only non clock INGEN inputs can be specified, here. Also, no INGEN input can be specified, which has not reached the final steady state, at the end of the previous simulation period.

B.4 GENERAL PROCEDURE:

With the input formatting as elaborated in above sections, the general procedure to be followed, is as described here.

i) The test circuit should be strictly an NMOS digital circuit, which can be described with the MOSIMR elements (Table B.1). There should be no wired connection of gate outputs, except those involving pass transistors.

ii) Visualise and draw the circuit diagram in terms of elements, supported by MOSIMR. Interconnections of these elements are the circuit nodes and interconnections of devices inside an element are the internal element nodes.

Number the circuit nodes, sequentially, starting from 1. Number elements of each type separately, starting from 1.

iii) If the test circuit is a repeatitive structure, where particular blocks can be identified to be connected regularly; each individual block can be described as a subckt. Each subckt. is an independent entity where the element and node numbering is local, done in the same way as in (ii), irrespective of the main circuit numbering. However, the nodes in a subckt. are numbered sequentially in such a way that internal nodes are numbered first and the external nodes (having connections with the environment) are numbered later, with higher integers than the internal nodes.

A subckt. may be located in different parts of the main circuit. It is CALL'ed the no. of times it occurs in the main circuit, supplying relevant external nodes (relevant with the main circuit. environment, where it is to be expanded) in each CALL. A subckt. cannot include another subckt.

iv) For proper linking of subckts. with main ckt. environment care must be taken in preparing the data in cards 7 and 9(c) in Sec. B.2.

Mark the variable LIN (card 7) with an 'L', in a main ckt. elem. data for the corresponding input/output node 'N' in the following cases -

a) If N is an output node of the element.

b) If N is an input node of the element and the element is the leading member of the NB chain, at N.

Include a subckt. element in EXTERN (card 9(c)), associated with a dummy external node N, only in the following cases -

a) N is an output node of the subckt. element.

b) N is an input node of the subckt. element and the element is the leading member of the NB chain, at N.

v) EPSLN is the lower limit, set for comparison of a node voltage in consecutive time steps. A change in node voltage V_n is recognised in the interval t_1 to t_2, if

$$|V_n(t_1) - V_n(t_2)| \geq \text{EPSLN},$$

else V_n is considered to be constant in that interval.

This EPSLN and the simulation time step DELTA, are the two parameters which can be varied to trade accuracy and numerical stability of analysis with speed.

As EPSLN ↗ SPEED ↗ and stability and accuracy ↘

As DELTA ↗ SPEED ↗ and stability and accuracy ↘

Default values of the parameters (appropriate for node capacitances of the order of .01 pF), EPSLN = .00001V and DELTA = .1 nsec. have been specified in the DATA statement, at the beginning of the main pgm.

If any change in these parameter values are to be made the program MOSIMR is copied and the DATA statement in the copy is modified.

vi) Inputs can be applied only at main circuit nodes. Nodes and elements to be monitored, must not belong to a subcircuit.

Table B.1:

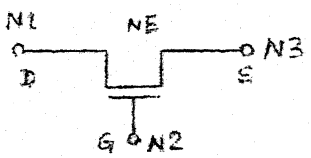
Name of elem. type	Implication	Element specification format
INVER	Inverter	

'INVER' NE

FORMAT(3X, A5, 1X, I3)

N1	N2	Name of NB elem. at N1	No.of NB elem. at N1	Name of of succ. elem.at N2	No. of succ. elem. at N2

FORMAT(3X,2(I3,IX),2(A5,1X,I3,1X)

IPAST	Pass transistor	

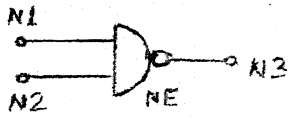
'IPAST' NE

FORMAT(3X,A5,1X,I3)

N1	N2	N3	Name of NB elem. at N1	No. of NB elem. at N1	Name of NB elem. at N2	No. of NB elem. at N2

Name of succ.elem. at N3	No. of succ. elem. at N3

FORMAT(3X,3(I3,1X),3(A5,1X,I3,1X)

Name of elem. type	Implication	Element specification format
NAND2	Two-input gate	

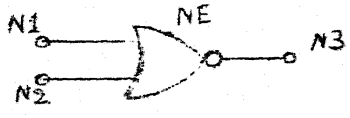
'NAND2' NE

FORMAT(3X,A5,1X,I3)

N1 N2 N3 Name of NB elem. at N1 | No. of NB elem. at N1 | Name of NB elem. at N2

No. of NB elem. at N2 | Name of NB succ. elem. at N3 | No. of succ. elem. at N3

FORMAT(3X,3(I3,1X),3(A5,1X,I3,1X))

NORG2	Two-input NOR gate	

'NORG2' NE

FORMAT (3X, A5, 1X, I3)

N1 N2 N3 Name of NB elem. at N1 | No. of NB elem. at N1 | Name of NB elem. at N2

No. of NB elem. at N2 | Name of succ. elem. at N3 | No. of succ. elem. at N3

FORMAT(3X,3(I3,1X),3(A5,1X,I3,1X))

Table B.1 contd...

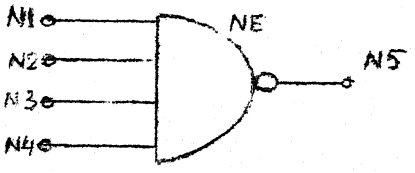
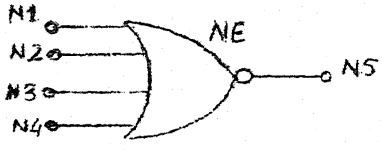
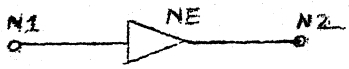
Name of elem. type	Implication	Element specification format
NAND4	Four-input NAND gate	 'NAND4' NE FORMAT (3X, A5, 1X, I3) N1 N2 N3 N4 N5 Name of NB elem. at N1 / No. of NB elem. at N1 ... Name of NB elem. at N4 / No. of NB elem. at N4 Name of succ. elem. at N5 / No. of succ. elem. at N5 FORMAT(3X,5(I3,1X),5(A5,1X,I3,1X))
NORG4	Four-input NOR gate	 'NORG4' NE FORMAT(3X,A5,1X,I3) N1 N2 N3 N4 N5 Name of NB elem. at N1 / No. of NB elem. at N1 ... Name of NB elem. at N4 / No. of NB elem at N4 Name of succ. elem. at N5 / No. of succ. elem. at N5

Table B.1 contd..

Name of elem. type	Implication	Element specification format

FORMAT(3X,5(I3,1X),5(A5,1X,I3,1X))

NIBUF Noninverting super buffer



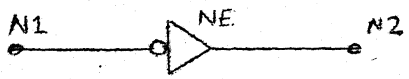
'NIBUF' NE

FORMAT (3X, A5, 1X, I3)

N1	N2	Name of NB elem. at N1	No. of NB elem. at N1	Name of succ. elem. at N2	No. of succ. elem. at N2

FORMAT(3X,2(I3,1X),2(A5,1X,I3),1X))

INBUF Inverting super buffer



'INBUF' NE

FORMAT (3X, A5, 1X, I3)

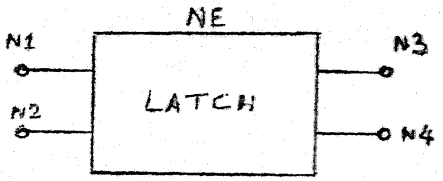
N1	N2	Name of NB elem. at N1	No. of NB elem. at N1	Name of succ.elem. at N2	No. of succ. elem. at N2

FORMAT (3X,2(I3,1X),2(A5,1X,I3,1X))

Table B.1 contd...

Name of elem. type	Implication	Element specification format

LATCH — NOR Gate S/R latch



'LATCH' NE

FORMAT (3X, A5, 1X, I3)

N1 N2 N3 N4 Name of NB elem. at N1 | No. of NB elem. at N1 | Name of NB elem. at N2

No. of NB elem. at N2 | Name of succ. elem.at N3 | No. of succ. elem.at N3

Name of succ. elem. at N4 | No. of succ. elem. at N4

FORMAT(3X,4(I3,1X),4((A5,1X,I3,1X))

NOELM — No element

'NOELM' 000

FORMAT (3X, A5, 1X, I3)

If a subckt. elem. has a main circuit element. as its NB/succ. or a main ckt. elem. has a subckt. elem. as its NB/succ., the corresponding linking is done during subckt. expansion, when the subckt is CALL'ed. But in the circuit description (the subckt. and main ckt. being independent) the corresponding NB/succ. field of data (cards 5 and 8) is filled with 'NOELM'.

N.B. NE = element no.

Table B.2: Input Table

General format for any kind of input : 'INPUT' NI
FORMAT(3X, A5, 1X, I3)

NA Input kind Name of succ. elem. No. of succ. elem.

FORMAT (3X,I3,1X,2(A5,1X),I3)

Input kind	Implication	Input waveform specifications
EXDEF	Externally defined input	The voltages at EXDEF input nodes are specified in INP.PRE in proper order, at each simulation time step, in the format FORMAT(3X,15(F7.3,1X) over the entire simulation period.
INGEN	Internally generated input Types are : i) REDGE(rising edge) ii) FEDGE(falling edge) iii) PPULS(positive pulse) iv) NPULS(negative pulse) v) CONST(constant)	Timings such as i) duration after which input commences to change ii) Rise/fall time iii) High/low period are fed-in interactively through TTY
CLOCK	single phase clock	
PHAS1/ PHAS2	Two phase clock	

N.B.: NI = Input no.
NA = Node where input is applied.

REFERENCES

[1] Carver Mead and Lynn Conway, Introduction to VLSI Systems. Addison-Wisely Publishing Company, 1980.

[2] W.N. Carr and J.P. Mize, MOS/LSI Design and Applications. N.Y. McGraw-Hill 1972 (Texas Instruments Electronics Series).

[3] W.M. Penney, MOS Integrated Circuits. Van Nostrand Reinhold Co. 1972.

[4] M.J. Howes and D.V. Morgan (editors), Large Scale Integration, John Wiley and Sons 1981.

[5] L.O. Chua and P.M. Lin, Computer Aided Analysis of Electronics Circuits. New Jersey, Prentice Hall, 1975.

[6] M.A. Breuer and A.D. Friedman, Diagnosis and Reliable Design of Digital Systems, Computer Science Press Inc., 1976.

[7] H. Schichman and D.A. Hodges, 'Modeling and Simulation of Insulated Gate Field-effect Transistor Switching Circuits', IEEE J. Solid State Circuits, vol. SC-3, pp. 285-289, Sept. 1968.

[8] SPICE Version 2G User's Guide, 10 Aug. 1981.

[9] 'LAMP', The Bell System Technical Journal, vol. 53, No. 8, pp. 1431-1554, Oct. 1974.

[10] B.R. Chawla, H.K. Gummel, and P. Kozak, 'MOTIS - an MOS Timing Simulator', IEEE Trans. Circuits Syst., vol. CAS-22, pp. 901-909, Dec. 1975.

[11] E. Lelarasmee, A.E. Ruehli, and A. Sangiovanni Vincentelli, 'The Waveform Relaxation Method for Time - Domain Analysis of Large Scale Integrated Circuits', IEEE Trans. Computer-Aided Design, vol. CAD-1, No. 3, pp. 131-145, July, 1982.

[12] A.R. Newton and A. Sangiovanni-Vincentelli, 'Relaxation - based Electric Simulation', IEEE Trans. Computer Aided Design, vol. CAD-3, No. 4, pp. 308-331, Oct. 1984.

[13] A.R. Newton, 'The Simulation of Large Scale Integrated Circuits', IEEE Trans. Circuits Syst., vol. CAS-26, pp. 741-749, Sept. 1979.

[14] G. De Micheli, A.R. Newton, and A. Sangiovanni Vincentelli, 'Symmetric Displacement Algorithms for the Timing Analysis of MOS VLSI Circuits', IEEE Trans. Computer-Aided Design, vol. CAP-3, pp. 167-180, July,1983.

[15] CAD-82, Conference Report, Brighton Metropole Sussex U.K. 30 March - 1 April,1982, Organised by the journal Computer-Aided Design.

[16] W.S. Dorn and D.D. McCracken,Numerical Methods with FORTRAN IV case Studies, John Wiley and Sons, 1972.

[17] DEC System 10 FORTRAN Programmer's Reference Manual. Digital Equipment Corporation, Maynard, Massachusetts.

[18] Narsingh Deo, Graph Theory, Prentice Hall of India Pvt. Ltd., 1984.

[19] E. Horowitz and S. Sahni, Fundamentals of Data Structures. Virginia, Goodland Hills, Computer Science Press,1976.